

**PREDICTING CLINICAL VARIABLES
FROM NEUROIMAGES
USING FUSED SPARSE GROUP LASSO**

by

Joanne C. Beer

MS, Oregon Health & Science University, 2013

Submitted to the Graduate Faculty of

Department of Biostatistics

Graduate School of Public Health in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

University of Pittsburgh

2018

UNIVERSITY OF PITTSBURGH

Graduate School of Public Health

This dissertation was presented

by

Joanne C. Beer

It was defended on

July 17, 2018

and approved by

Dissertation Co-Advisor:
Stewart J. Anderson, PhD
Professor

Departments of Biostatistics and Clinical and Translational Science
Graduate School of Public Health
University of Pittsburgh

Dissertation Co-Advisor:
Robert T. Krafty, PhD
Associate Professor
Department of Biostatistics
Graduate School of Public Health
University of Pittsburgh

Committee Member:
Dana Tudorascu, PhD
Associate Professor
Departments of Medicine, Biostatistics, Psychiatry, and Clinical and Translational Science
School of Medicine
University of Pittsburgh

Committee Member:
Howard J. Aizenstein, MD, PhD
Professor
Departments of Psychiatry, Bioengineering, and Clinical and Translational Science
School of Medicine
University of Pittsburgh

Copyright © by Joanne C. Beer
2018

**PREDICTING CLINICAL VARIABLES
FROM NEUROIMAGES
USING FUSED SPARSE GROUP LASSO**

Joanne C. Beer, PhD

University of Pittsburgh, 2018

ABSTRACT

Predictive models in which neuroimage features serve as predictors and a clinical variable is modeled as the outcome are good candidates for clinical application because (1) they can exploit dependencies between predictor variables and thus potentially explain more variability in the outcome than a mass univariate approach, and (2) they allow inference at the individual level, such that a prediction can be obtained for a new individual whose data was not used to train the model. This dissertation proposes methods for neuroimaging predictive models that not only aim for prediction accuracy, but also seek interpretability and potential insight into the underlying pathophysiology of neuropsychiatric disorders.

In the first part of this dissertation we propose the fused sparse group lasso penalty, which encourages structured, sparse, interpretable solutions by incorporating prior information about spatial and group structure among voxels. We derive optimization steps for fused sparse group lasso penalized regression using the alternating direction method of multipliers algorithm. With simulation studies, we demonstrate conditions under which fusion and group penalties together outperform either of them alone. We then use fused sparse group lasso to predict continuous measures from resting state magnetic resonance imaging data using the Autism Brain Imaging Data Exchange dataset. In the second part of this dissertation we use fused sparse group lasso to predict age from multimodal neuroimaging data in a sample of cognitively normal adults aged 65 and older. In general, we show how the incorporation of

prior information via the fused sparse group lasso penalty can enhance the interpretability of neuroimaging predictive models while also yielding good prediction performance.

Public health significance: Psychiatric disorders and neurological diseases such as Alzheimer’s present a large public health burden. As of yet, there have been relatively few translations of basic neuroscience findings to clinical applications in psychiatry. Prediction models using neuroimaging data can potentially help clinicians with diagnosis and prediction of prognosis and treatment response. Establishing interpretable neuroimaging-based biomarkers can improve our understanding of the neurobiological mechanisms underlying neuropsychiatric disorders and suggest approaches for prevention and treatment.

TABLE OF CONTENTS

1.0 INTRODUCTION	1
2.0 INCORPORATING PRIOR INFORMATION WITH FUSED SPARSE GROUP LASSO	4
2.1 Introduction	4
2.2 Fused sparse group lasso	7
2.2.1 Model	7
2.2.2 Estimator	7
2.2.3 Optimization algorithm	9
2.2.3.1 Stopping criteria	13
2.2.3.2 Adaptive step-size	13
2.2.4 Adaptive fused sparse group lasso	14
2.3 Simulation study	14
2.3.1 Simulation study methods	14
2.3.2 Simulation study results	17
2.4 Application to ABIDE neuroimaging data	24
2.4.1 Application methods	25
2.4.2 Application results	27
2.5 Conclusions	35
3.0 FUSED SPARSE GROUP LASSO FOR MULTIMODAL NEUROIMAGING PREDICTION	36
3.1 Introduction	36
3.1.1 Age-related structural and functional changes in the brain	37

3.1.2	Predicting age from neuroimages	38
3.1.3	Study objectives	40
3.2	Multimodal fused sparse group lasso	41
3.2.1	Model	41
3.2.2	Estimator	42
3.3	Application to Normal Aging multimodal neuroimaging data	44
3.3.1	Normal Aging dataset	44
3.3.2	Methods	46
3.3.2.1	MRI data collection	46
3.3.2.2	MRI data preprocessing	46
3.3.2.3	Feature screening	47
3.3.2.4	Unimodal prediction models	48
3.3.2.5	Multimodal prediction models	50
3.3.3	Results	50
3.3.3.1	Feature screening	50
3.3.3.2	Unimodal prediction models	52
3.3.3.3	Multimodal prediction models	57
3.3.3.4	Analysis of brain-PAD from best models	60
3.4	Discussion	62
4.0	CONCLUSIONS	66
APPENDIX A.	SIMULATION STUDY RESULTS	72
APPENDIX B.	ABIDE APPLICATION RESULTS	82
APPENDIX C.	NORMAL AGING STUDY RESULTS	85
APPENDIX D.	R CODE	97
D.1	R function: <code>softthresh.R</code>	97
D.2	R function: <code>makeKmatrix2d.R</code>	98
D.3	R function: <code>makeKmatrix3d.R</code>	99
D.4	R function: <code>fsgl.fit.R</code>	100
D.5	R function: <code>fsgl.cv.R</code>	102
APPENDIX E.	MATLAB CODE	104

E.1	MATLAB function:	<code>makeKmatrix.m</code>	104
E.2	MATLAB function:	<code>makeKmatrix_adaptive.m</code>	106
E.3	MATLAB function:	<code>fsglfit.m</code>	108
E.4	MATLAB function:	<code>fsglfit_adaptive.m</code>	110
BIBLIOGRAPHY			112

LIST OF TABLES

2.1	Examples of unstructured and structured penalty terms	5
2.2	Optimal (α, γ) for each simulation scenario	18
2.3	Comparison of estimators applied to ABIDE dataset	33
3.1	Normal Aging dataset descriptive summary	45
3.2	Comparison of estimators applied to Normal Aging dataset: GMD data . . .	53
3.3	Comparison of estimators applied to Normal Aging dataset: DMN/ASN data	54
3.4	Comparison of estimators applied to Normal Aging dataset: Multimodal GMD and DMN/ASN data	58
A1	Simulation results scenario 1A – Completely aggregated	73
A2	Simulation results scenario 2B – Partially aggregated	74
A3	Simulation results scenario 3C – Completely distributed	75
A4	Simulation results scenario 4A – Sparse completely aggregated	76
A5	Simulation results scenario 5B – Sparse partially aggregated	77
A6	Simulation results scenario 6C – Sparse completely distributed	78
A7	Simulation results scenario 7B – Extra sparse partially aggregated	79
A8	Simulation results scenario 8B – Misspecified partially aggregated	80
A9	Simulation results scenario 9B – Misspecified sparse partially distributed . . .	81
B1	ABIDE dataset descriptive summary	83
B2	Linear regression model for adjusting Social Responsiveness Scale scores . . .	84
C1	Univariate feature screening of GMD data for predicting age	86
C2	Univariate feature screening of DMN/ASN data for predicting age	87

LIST OF FIGURES

2.1	Penalty term subsets for fused sparse group lasso produced by different (α, γ) combinations	10
2.2	Example of \mathbf{K} matrix for fused sparse group lasso penalty	12
2.3	Simulation study group structures	15
2.4	Simulation study true coefficients	16
2.5	Simulation study results for true coefficients 1A, 2B, and 3C	19
2.6	Simulation study results for true coefficients 4A, 5B, and 6C	20
2.7	Simulation study results for true coefficients 7B, 8B, and 9B	21
2.8	Simulation study squared bias, variance, and MSE of predicted responses . .	22
2.9	Example set of estimated coefficients for one simulation iteration of scenario 2B	23
2.10	Overview of ABIDE application methods	28
2.11	Fused sparse group lasso regression applied to ABIDE dataset, methods . . .	29
2.12	Fused sparse group lasso regression applied to ABIDE dataset, results	32
2.13	ABIDE dataset estimated coefficients	34
3.1	Possible variations of the fused sparse group lasso penalty for multimodal data	43
3.2	Age distribution for Normal Aging dataset	47
3.3	Overview of Normal Aging study data analysis methods	48
3.4	Feature screening results for GMD and DMN/ASN data	51
3.5	Estimated coefficients for best GMD unimodal model	55
3.6	Estimated coefficients for best DMN/ASN unimodal model	56
3.7	Predicted versus actual age for best unimodal and multimodal models	59
3.8	Correlation of age-adjusted brain-PAD for best unimodal models	60

3.9	Brain-PAD versus actual age for best unimodal and multimodal models . . .	61
C1	Normal Aging unimodal GMD cross-validation error curves	88
C2	Normal Aging unimodal DMN/ASN cross-validation error curves	89
C3	Normal Aging multimodal GMD and DMN/ASN cross-validation error curves	90
C4	Normal Aging unimodal GMD estimated coefficients	91
C5	Normal Aging unimodal DMN/ASN estimated coefficients	92
C6	Normal Aging multimodal GMD and DMN/ASN estimated coefficients . . .	93
C7	Normal Aging unimodal GMD predicted versus actual age	94
C8	Normal Aging unimodal DMN/ASN predicted versus actual age	95
C9	Normal Aging multimodal GMD and DMN/ASN predicted versus actual age	96

1.0 INTRODUCTION

The first successful functional magnetic resonance imaging (fMRI) studies of the human brain were carried out in early 1991 ([Bandettini 2012](#)). Since then, there have been relatively few translations of basic neuroscience findings to clinical applications in psychiatry, such as the use of brain biomarkers for determining diagnosis, prognosis, or predicting treatment response ([Kapur et al. 2012](#), [Woo et al. 2017](#)). The traditional mass univariate approach in neuroimaging, which fits a model to each voxel independently, has been successful at identifying group-level brain structure and function. However, a predictive model approach, in which neuroimage features serve as predictors and a clinical variable is modeled as the outcome, may be better suited to clinical application. Predictive models are able to exploit dependencies between brain regions and thus can potentially explain more variability in the outcome than a mass univariate approach. Moreover, predictive models allow inference at the individual level such that a prediction can be obtained for a new individual whose data was not used to train the model.

Some neuroimaging-based predictive models have aimed solely for prediction accuracy, but failed to provide insight into the underlying pathophysiology because they modeled nonlinearities and interactions using complex machine learning algorithms, or ultimately relied on signal originating in head motion or eye blinks which happened to be correlated with the outcome of interest ([Woo et al. 2017](#)). Ideally we would like to achieve both accuracy and interpretability with a predictive model, but sometimes we might be willing to trade some measure of prediction accuracy for interpretability. It is possible that, by constraining the parameter solution space using well-chosen prior information, we can achieve parameter estimates that are more neuroscientifically informative, without sacrificing much or perhaps even gaining in prediction accuracy.

In this dissertation, we show how the fused sparse group lasso, a structured, sparse estimator, can incorporate prior information into a predictive model, thereby allowing researchers to harness results from the extensive recent research on brain structural and functional connectivity. Our goals include not only prediction accuracy, gauged by how well the model predicts the response for independent test data, but also interpretable parameter estimates, as based on the following criteria: (1) Model structure entails that parameter values have a straightforward meaning; e.g., linear models tend to be more interpretable than non-linear models. (2) Models are appropriately sparse, including only relevant predictors, while not excluding any relevant predictors. (3) Parameter estimates are understandable in light of existing background knowledge. In a translational neuroimaging context, this would mean that the brain regions implicated by the model estimates are neuroscientifically plausible according to existing knowledge or provide new insight into the neurobiological mechanism influencing the clinical outcome, and can potentially be used to establish biomarkers.

While interest in predicting continuous outcomes is increasing, the majority of work in prediction from neuroimages has focused on classification problems ([Cohen et al. 2011](#), [Arbabshirani et al. 2017](#)). Numerous studies have built classifiers to differentiate patients from healthy controls or sort patients into diagnostic groups on the basis of neuroimaging data. However, many psychiatric diagnostic categories are of questionable validity and may group together individuals with heterogenous etiology underlying their symptoms. The inadequacy of current diagnostic categories was a primary motivator for the National Institute of Mental Health’s Research Domain Criteria initiative, a research framework that encourages a dimensional approach to human behavior through the investigation of constructs that cut across diagnostic categories ([Insel et al. 2010](#)). Continuous measures of more fundamental phenotypic traits may map better onto underlying neurobiology and may be particularly suitable for spectrum disorders such as autism. While our proposed penalty could easily be extended for use in conjunction with a variety of loss functions, including those for categorical outcomes such as logistic loss, here we adopt a dimensional approach using continuous outcomes.

For neuroimaging applications, spatial regularization using the ℓ_1 or ℓ_2 norm of the image gradient has shown good performance, yielding smooth solutions in spatially contiguous brain

regions. However, recently enormous resources have been devoted to establishing structural and functional brain connectivity networks, yielding information which can be used to define spatially distributed yet related groups of voxels. In Chapter 2 we propose a penalty — fused sparse group lasso — which can exploit prior information about spatial and group structure among voxels, thereby encouraging structured, sparse, interpretable solutions. We outline an optimization algorithm based on the alternating direction method of multipliers (ADMM) to fit the fused sparse group lasso estimator (Section 2.2.3), and demonstrate properties of the estimator in a simulation study (Section 2.3) and in application to resting state functional magnetic resonance imaging data from the Autism Brain Imaging Data Exchange (ABIDE) dataset (Section 2.4).

Multimodal neuroimaging studies that combine data from different neuroimaging modalities, such as structural and functional MRI, have become increasingly popular in recent years. Since multimodal models take advantage of the complementary strengths of different neuroimaging modalities, they often yield better classification or prediction performance than unimodal models (Liu et al. 2015, Calhoun & Sui 2016). In Chapter 3 we explore ways in which fused sparse group lasso can incorporate complementary information derived from different neuroimaging modalities. We use fused sparse group lasso to predict age from multimodal imaging data in a sample of cognitively normal adults aged 65 and older. Finally, in Chapter 4 we conclude with an evaluation of how the fused sparse group lasso penalty can potentially enhance the interpretability of neuroimaging predictive models, its prediction performance, and discuss challenges and areas for further study.

2.0 INCORPORATING PRIOR INFORMATION WITH FUSED SPARSE GROUP LASSO

2.1 INTRODUCTION

Consider the linear regression model

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}, \tag{2.1}$$

where $\mathbf{y} \in \mathbb{R}^n$ is a continuous outcome (e.g. score on a clinical depression rating scale), $\mathbf{X} \in \mathbb{R}^{n \times p}$ is a predictor matrix (e.g. neuroimage voxel values), $\boldsymbol{\beta} \in \mathbb{R}^p$ is an unknown vector of coefficients, $\boldsymbol{\epsilon} \in \mathbb{R}^n$ is the error, $E(\boldsymbol{\epsilon}) = \mathbf{0}$, and $E(\boldsymbol{\epsilon}^T \boldsymbol{\epsilon}) = \sigma^2 \mathbf{I}_n$. This represents a high dimensional setting where the number of subjects n is much less than the number of predictors p , which can be on the order of 100,000 voxels. To obtain a unique solution for $\boldsymbol{\beta}$, we can constrain the optimization problem using the penalized least squares estimator

$$\hat{\boldsymbol{\beta}} = \arg \min_{\boldsymbol{\beta} \in \mathbb{R}^p} \frac{1}{2} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda J(\boldsymbol{\beta}), \tag{2.2}$$

where $\lambda \geq 0$ is a tuning parameter controlling the level of regularization. The penalty term $J(\boldsymbol{\beta})$ can impose both sparsity and structure, thereby constraining the solution space according to *a priori* knowledge about relationships between elements of $\boldsymbol{\beta}$. In a neuroimaging context, for example, this information might include spatial proximity or previously established functional connectivity networks. Examples of unstructured and structured penalties are presented in Table 2.1.

Simulation studies and applications to real neuroimaging datasets (mostly using functional magnetic resonance imaging (fMRI)) have shown that penalties enforcing spatial smoothness frequently outperform unstructured penalties (Michel et al. 2011, Baldassarre

Table 2.1: Examples of unstructured and structured penalty terms

Unstructured penalties	$J(\beta)$
Lasso (Tibshirani 1996)	$\ \beta\ _1$
Ridge (Hoerl & Kennard 1970)	$\ \beta\ _2^2$
Elastic net (Zou & Hastie 2005)	$\alpha\ \beta\ _1 + (1 - \alpha)\ \beta\ _2^2; \alpha \in [0, 1]$
Structured penalties	$J(\beta)$
Isotropic total variation (Rudin et al. 1992)	$\ \mathbf{D}\beta\ _{2,1};$ matrix \mathbf{D} encodes spatial structure
Fused lasso* (Tibshirani et al. 2005)	$\alpha\ \beta\ _1 + (1 - \alpha)\ \mathbf{D}\beta\ _1; \alpha \in [0, 1]$
Graph net** (Grosenick et al. 2013)	$\alpha\ \beta\ _1 + (1 - \alpha)\ \mathbf{D}\beta\ _2^2; \alpha \in [0, 1]$
Group lasso (Yuan & Lin 2006)	$\sum_{g \in \mathcal{G}} \sqrt{p_g} \ \beta_g\ _2; \text{groups } \mathcal{G} \text{ form a partition of } \beta$
Sparse group lasso (Simon et al. 2013)	$\alpha\ \beta\ _1 + (1 - \alpha) \sum_{g \in \mathcal{G}} \sqrt{p_g} \ \beta_g\ _2; \alpha \in [0, 1]$

*Also known as anisotropic total variation- ℓ_1

**Also known as sparse graph Laplacian

et al. 2012, Gramfort et al. 2013, Grosenick et al. 2013, Fiot et al. 2014, Xin et al. 2014). Not only do spatially-informed penalties yield more interpretable estimates insofar as they select contiguous groups of voxels in neuroscientifically plausible brain regions, but they often show better prediction performance. For example, Michel et al. (2011) found that the isotropic total variation penalty gave higher prediction accuracy and recovered the support of the true parameters better than elastic net and linear support vector regression in simulation studies, and for real fMRI data produced estimates that were less dispersed and more neuroscientifically interpretable. Baldassarre et al. (2012) reported that the anisotropic total variation- ℓ_1 penalty (equivalent to a three dimensional fused lasso) had the highest classification accuracy in a real fMRI data application as compared with lasso, elastic net, graph net, and anisotropic total variation alone. Additionally, it produced the most stable estimates across validation folds. Both of these studies involved healthy participants viewing various images in the fMRI scanner, and models were designed to predict or classify some feature of the images. In a more clinically-oriented study, Fiot et al. (2014) used structural neuroimaging data to predict Alzheimer’s disease progression. The authors compared several penalties, including ridge, lasso, elastic net, non-sparse graph net, isotropic total variation, and isotropic total variation- ℓ_1 . The spatially-informed penalties yielded more neuroscientifically relevant

coefficient maps and statistically better classification accuracy than the unstructured penalties.

In addition to spatial regularization, group-structured regularization has shown promise in predictive neuroimaging models. Shimizu et al. (2015) compared logistic regression using lasso, group lasso, and sparse group lasso penalties, linear support vector machine (SVM), and random forest for classifying depression patients and healthy controls based on fMRI data. For the group lasso penalties, voxels were grouped according to known functional and anatomical brain regions. The authors found that group lasso and sparse group lasso were superior to lasso and random forest and comparable to SVM in terms of classification accuracy, but unlike SVM, they produced sparse and more interpretable models. Rather than defining voxel groups *a priori*, Liu et al. (2014) used a data-driven agglomerative hierarchical clustering method to create a tree-structured grouping of voxels in grey matter density brain maps. Feature selection was then performed using a tree-structured group lasso penalty, and the selected features were used in a linear SVM to discriminate Alzheimer’s disease patients from healthy controls. The proposed method achieved higher classification accuracy than feature selection using the ℓ_1 lasso penalty or an anatomically-defined group lasso penalty.

For neuroimaging applications, we aim to incorporate two types of structure into the penalty term $J(\beta)$ of the estimator in Equation (2.2): (1) local spatial information, to encourage smooth coefficient estimates across neighboring voxels; and (2) spatially distributed groups, such as those defined by functional or structural networks or anatomical regions, to allow voxels within the same group to be selected or shrunk to zero together. We achieve this by combining ℓ_1 , fusion, and group lasso penalties into a fused sparse group lasso penalty. We found one instance of this penalty in the literature, in a multi-task learning context where groups consist of repeated measures of the same task and smoothing is applied across time points within a group (Zhou et al. 2012). To our knowledge, the fused sparse group lasso penalty has never been studied via simulations or used in a predictive model with neuroimaging data.

In the remainder of this chapter, we present the fused sparse group lasso estimator in Section 2.2 and derive update steps to fit the fused sparse group lasso penalized least squares regression model using the alternating direction method of multipliers algorithm in Section

2.2.3. We report methods and results of a simulation study in Section 2.3 and apply our method to resting state fMRI data from the Autism Brain Imaging Data Exchange repository in Section 2.4. We make concluding remarks in Section 2.5. We provide R and MATLAB functions for fitting the fused sparse group lasso estimator in Appendices D and E, and online at <https://github.com/jcbeer/fsgl>.

2.2 FUSED SPARSE GROUP LASSO

2.2.1 Model

Suppose we observe $\{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$ from n independent subjects, indexed by $i = 1, \dots, n$, where $y_i \in \mathbb{R}$ and $\mathbf{x}_i \in \mathbb{R}^p$. In the neuroimaging context considered here, y_i is a continuous scalar outcome for each subject such as age, depression scale or cognitive test score, and \mathbf{x}_i is a vector of voxel values from a three dimensional brain image such that each element of \mathbf{x}_i corresponds to one of p voxels. Assume that $\mathbf{y} = (y_1, \dots, y_n)^T$ and the columns of the matrix $\mathbf{X} = (\mathbf{x}_1 | \dots | \mathbf{x}_n)$ are centered, so we do not have an intercept term. Furthermore, we standardize the columns of \mathbf{X} to have standard deviation of one. We model the continuous outcome using standard linear regression as expressed in Equation (2.1).

2.2.2 Estimator

Since the number of voxels is typically orders of magnitude larger than the number of subjects, i.e., $p \gg n$, regularization is required to obtain a unique solution for $\boldsymbol{\beta}$. We propose estimating $\boldsymbol{\beta}$ by minimizing the sum of the loss function and three penalty terms:

$$\hat{\boldsymbol{\beta}} = \arg \min_{\boldsymbol{\beta} \in \mathbb{R}^p} L(\boldsymbol{\beta}) + \lambda_1 \|\boldsymbol{\beta}\|_1 + \lambda_2 \|\mathbf{D}\boldsymbol{\beta}\|_1 + \lambda_3 \Omega^{\mathcal{G}}(\boldsymbol{\beta}); \quad (2.3)$$

where $L(\boldsymbol{\beta})$ is the loss function (e.g., least squares); $\|\boldsymbol{\beta}\|_1 = \sum_{j=1}^p |\beta_j|$ is the ℓ_1 norm of $\boldsymbol{\beta}$; $\mathbf{D}_{m \times p}$ is the three dimensional fusion matrix for fused lasso, e.g., for a $2 \times 2 \times 2$ cubic image,

$$\mathbf{D} = \begin{bmatrix} 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \\ 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 \end{bmatrix},$$

and $\|\mathbf{D}\boldsymbol{\beta}\|_1$ is the fusion penalty; $\Omega^{\mathcal{G}}(\boldsymbol{\beta}) = \sum_{g \in \mathcal{G}} \sqrt{p_g} \|\boldsymbol{\beta}_g\|_2$ is the $\ell_{2,1}$ group lasso penalty, which applies the ℓ_2 norm, $\|\boldsymbol{\beta}_g\|_2 = \sqrt{\boldsymbol{\beta}_g^T \boldsymbol{\beta}_g}$, to the coefficients $\boldsymbol{\beta}_g$ for each group $g \in \mathcal{G}$, each of size p_g ; and $\lambda_1, \lambda_2, \lambda_3 \geq 0$ are regularization tuning parameters.

The three penalty terms incorporate prior information into the estimator, encouraging the solution to have both sparsity and a particular structure. The standard lasso ℓ_1 penalty encourages overall sparsity. The fusion penalty penalizes the absolute differences between coefficients at neighboring voxels, thereby encouraging local smoothness. The group lasso penalty encourages a group-level structure; entire groups may be selected or shrunk to zero together. For example, if groups are defined by functional networks, the penalty allows voxels involved in a common network to be shrunk to zero if that network is not important for prediction. Given the overlapping structure of brain networks, overlapping groups are another possibility worth considering. With appropriate weighting and a latent variable approach ([Jacob et al. 2009](#), [Obozinski et al. 2011](#)), the estimator could also accommodate overlapping groups.

For ease of selecting values for the tuning parameters via cross-validation, it is convenient to reparameterize (2.3) as follows:

$$\hat{\boldsymbol{\beta}} = \arg \min_{\boldsymbol{\beta} \in \mathbb{R}^p} L(\boldsymbol{\beta}) + \alpha\gamma\lambda \|\boldsymbol{\beta}\|_1 + (1 - \gamma)\lambda \|\mathbf{D}\boldsymbol{\beta}\|_1 + (1 - \alpha)\gamma\lambda \Omega^g(\boldsymbol{\beta}), \quad (2.4)$$

such that $\lambda > 0$ controls the overall level of regularization, $\alpha \in [0, 1]$ controls the balance between the two sparsity inducing penalties (lasso and group lasso), and $\gamma \in [0, 1]$ controls the balance between the two sparsity inducing penalties and the fusion penalty. When $\alpha = 1$ and $\gamma = 1$, the estimator reduces to the standard lasso; when $\alpha = 0$ and $\gamma = 1$, the estimator reduces to the group lasso, and so on for other subsets of the three penalty terms (Figure 2.1).

2.2.3 Optimization algorithm

While the optimization problem (2.3) is convex for convex loss functions, due to the fusion penalty term, it is non-separable across groups of $\boldsymbol{\beta}$, so block-wise gradient descent strategies often employed for group lasso are not directly applicable. However, one algorithm that works in this case is the alternating direction method of multipliers (ADMM, Boyd et al. (2011)).

For simplicity, we assume that the groups are non-overlapping and form a partition of $\boldsymbol{\beta}$, so that each coefficient belongs to exactly one group. For applying ADMM, we follow a strategy similar to that employed in Huo & Tseng (2017) and exploit the fact that $|\beta_j| = \sqrt{\beta_j^2}$ and $|\beta_j - \beta_{j-1}| = \sqrt{(\beta_j - \beta_{j-1})^2}$. Then we can reformulate the lasso and fusion ℓ_1 penalty terms as sets of $\ell_{2,1}$ group penalties whose groups have only one member. The reformulated penalty has an overlapping group structure, with each coefficient belonging to both its own one-member group and one of the groups forming the partition of $\boldsymbol{\beta}$, and additionally we treat each absolute difference specified by the fusion matrix \mathbf{D} as a group. If there are p coefficients, \mathbf{D} has m rows, and there are G groups that form a partition of $\boldsymbol{\beta}$, then the total number of effective groups is $p + m + G = N$.

Using a least-squares loss function, we now write the objective function (2.3) as

$$\arg \min_{\boldsymbol{\beta} \in \mathbb{R}^p} \frac{1}{2} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \sum_{j=1}^N \lambda_j w_j \|\mathbf{K}_j \boldsymbol{\beta}\|_2, \quad (2.5)$$

			GAMMA (γ)		
			\leftarrow more smoothing / less sparsity less smoothing / more sparsity \rightarrow		
			$\gamma = 0$	$\gamma \in (0, 1)$	$\gamma = 1$
ALPHA (α)	\leftarrow less grouping / more L1	$\alpha = 0$	Fusion	Fusion + Group	Group
	more grouping / less L1 \rightarrow	$\alpha \in (0, 1)$		Fusion + Group + L1	Group + L1
		$\alpha = 1$		Fusion + L1	Lasso (L1)

Figure 2.1: Penalty term subsets for fused sparse group lasso produced by different (α, γ) combinations

with

$$\{\lambda_j, \mathbf{K}_j\} = \begin{cases} \{\lambda_1, \mathbf{j}_j\} & \text{if } j \in \{1, 2, \dots, p\} \\ \{\lambda_2, \mathbf{d}_j\} & \text{if } j \in \{p+1, p+2, \dots, p+m\} \\ \{\lambda_3, \mathbf{G}_j\} & \text{if } j \in \{p+m+1, p+m+2, \dots, p+m+G\}, \end{cases}$$

where $\lambda_j \in \{\lambda_1, \lambda_2, \lambda_3\}$ are the regularization parameters for the lasso, fusion, and group lasso penalties, respectively; w_j are group weights (for group lasso typically $w_j = \sqrt{p_j}$ where p_j is the number of elements in group j); $\mathbf{j}_j \in \mathbb{R}^p$ corresponds to the j th row of the $p \times p$ identity matrix; $\mathbf{d}_j \in \mathbb{R}^p$ corresponds to the $(j-p)$ th row of the fusion matrix \mathbf{D} in the three dimensional fusion penalty; and $\mathbf{G}_j \in \mathbb{R}^{p_j \times p}$ is a sparse matrix where each row has a 1 at a column position corresponding to a member of group j .

For ADMM, we introduce the auxiliary variables $\boldsymbol{\theta}_j = \mathbf{K}_j \boldsymbol{\beta}$. The optimization problem becomes

$$\begin{aligned} & \text{minimize} \quad \frac{1}{2} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \sum_{j=1}^N \lambda_j w_j \|\boldsymbol{\theta}_j\|_2, \\ & \text{subject to} \quad \boldsymbol{\theta}_j - \mathbf{K}_j \boldsymbol{\beta} = \mathbf{0} \quad \text{for } j \in \{1, 2, \dots, N\}. \end{aligned}$$

Let $\mathbf{K} = (\mathbf{K}_1 | \dots | \mathbf{K}_N)$ (see Figure 2.2 for example), $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \dots, \boldsymbol{\theta}_N)^T$, and $\boldsymbol{\mu} = (\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_N)^T$.

The augmented Lagrangian is

$$\mathcal{L}_\rho(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\mu}) = \frac{1}{2} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \sum_{j=1}^N \lambda_j w_j \|\boldsymbol{\theta}_j\|_2 + \sum_{j=1}^N \left\{ \boldsymbol{\mu}_j^T (\boldsymbol{\theta}_j - \mathbf{K}_j \boldsymbol{\beta}) + \frac{\rho}{2} \|\boldsymbol{\theta}_j - \mathbf{K}_j \boldsymbol{\beta}\|_2^2 \right\}$$

where $\rho > 0$ is the step-size parameter and $\boldsymbol{\mu}_j$ are the dual variables for ADMM. After initialization of $\boldsymbol{\beta}$, $\boldsymbol{\theta}$, and $\boldsymbol{\mu}$, the update steps for ADMM consist of the following:

$$\begin{aligned} \boldsymbol{\beta}^{t+1} &= \arg \min_{\boldsymbol{\beta} \in \mathbb{R}^p} \mathcal{L}_\rho(\boldsymbol{\beta}, \boldsymbol{\theta}^t, \boldsymbol{\mu}^t) \\ \boldsymbol{\theta}_j^{t+1} &= \arg \min_{\boldsymbol{\theta}_j \in \mathbb{R}^{p_j}} \mathcal{L}_\rho(\boldsymbol{\beta}^{t+1}, \boldsymbol{\theta}, \boldsymbol{\mu}^t) \\ \boldsymbol{\mu}_j^{t+1} &= \boldsymbol{\mu}^t + \rho(\boldsymbol{\theta}_j^{t+1} - \mathbf{K}_j \boldsymbol{\beta}^{t+1}) \end{aligned}$$

For $\boldsymbol{\beta}^{t+1}$ and $\boldsymbol{\theta}_j^{t+1}$ updates, the corresponding \mathcal{L}_ρ subgradient will equal zero at the optimal solution. Thus, the update for $\boldsymbol{\beta}$ is

$$\boldsymbol{\beta}^{t+1} = (\mathbf{X}^T \mathbf{X} + \rho \mathbf{K}^T \mathbf{K})^{-1} (\mathbf{X}^T \mathbf{Y} + \mathbf{K}^T (\boldsymbol{\mu}^t + \rho \boldsymbol{\theta}^t)).$$

$$\mathbf{K} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & -1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

\mathbf{J} = identity matrix
for ℓ_1 lasso
penalty

\mathbf{D} = difference
matrix for fusion
penalty

Group 1

Group 2

}

\mathbf{G} = matrix
for group
penalty

Figure 2.2: Example of \mathbf{K} matrix for fused sparse group lasso penalty

The subgradient with respect to $\boldsymbol{\theta}_j$ is

$$\frac{\partial \mathcal{L}_\rho}{\partial \boldsymbol{\theta}_j} = \lambda_j w_j \frac{\partial \|\boldsymbol{\theta}_j\|_2}{\partial \boldsymbol{\theta}_j} + \boldsymbol{\mu}_j + \rho (\boldsymbol{\theta}_j - \mathbf{K}_j \boldsymbol{\beta}).$$

In general, the subgradient of the ℓ_2 -norm $\|\mathbf{q}\|_2$ is $\mathbf{q}/\|\mathbf{q}\|_2$ if $\mathbf{q} \neq \mathbf{0}$ and $\{\mathbf{r} \mid \|\mathbf{r}\|_2 \leq 1\}$ if $\mathbf{q} = \mathbf{0}$. Therefore, if $\boldsymbol{\theta}_j \neq \mathbf{0}$, the condition $\partial \mathcal{L}_\rho^{\boldsymbol{\theta}_j} / \partial \boldsymbol{\theta}_j = \mathbf{0}$ implies that

$$\boldsymbol{\theta}_j \left(1 + \frac{\lambda_j w_j}{\rho \|\boldsymbol{\theta}_j\|_2} \right) = \mathbf{K}_j \boldsymbol{\beta} - \frac{\boldsymbol{\mu}_j}{\rho}.$$

Let $\boldsymbol{\eta}_j = \mathbf{K}_j \boldsymbol{\beta} - \frac{\boldsymbol{\mu}_j}{\rho}$. The solution can be written in terms of the vector soft-thresholding operator $\mathcal{S}_\kappa(\mathbf{a}) = (1 - \kappa/\|\mathbf{a}\|_2)_+ \mathbf{a}$, where $\mathcal{S}_\kappa(\mathbf{0}) = \mathbf{0}$ and $(\cdot)_+ = \max(0, \cdot)$:

$$\begin{aligned}
\boldsymbol{\theta}_j^{t+1} &= \mathcal{S}_{1/\rho}(\boldsymbol{\eta}_j) \\
&= \left(1 - \frac{\lambda_j w_j}{\rho \|\boldsymbol{\eta}_j\|_2} \right)_+ \boldsymbol{\eta}_j.
\end{aligned}$$

2.2.3.1 Stopping criteria We use the stopping criteria described in [Boyd et al. \(2011\)](#). The algorithm terminates when the primal and dual residuals are small enough to achieve a linear combination of preselected levels of absolute (ϵ_{abs}) and relative (ϵ_{rel}) tolerance. Suitable values for ϵ_{abs} and ϵ_{rel} will depend on the specific application and scale of the data. Let the primal and dual residuals at iteration t be denoted as $r^t = \boldsymbol{\theta}^t - \mathbf{K}\boldsymbol{\beta}^t$ and $s^t = \rho \mathbf{K}^T(\boldsymbol{\theta}^t - \boldsymbol{\theta}^{t-1})$, respectively. The stopping criteria are $\|r^t\|_2 \leq \epsilon_{\text{pri}}^t$ and $\|s^t\|_2 \leq \epsilon_{\text{dual}}^t$, where

$$\epsilon_{\text{pri}}^t = \sqrt{p} \epsilon_{\text{abs}} + \epsilon_{\text{rel}} \max\{\|\mathbf{K}\boldsymbol{\beta}^t\|_2, \|\boldsymbol{\theta}^t\|_2\},$$

$$\epsilon_{\text{dual}}^t = \sqrt{|\boldsymbol{\theta}^t|} \epsilon_{\text{abs}} + \epsilon_{\text{rel}} \|\mathbf{K}^T \boldsymbol{\mu}^t\|_2,$$

and $|\boldsymbol{\theta}^t|$ represents the number of elements in $\boldsymbol{\theta}^t$.

2.2.3.2 Adaptive step-size To accelerate the convergence of the ADMM algorithm, we implement an adaptive step-size, ρ , following the procedure proposed by [He et al. \(2000\)](#) and implemented in [Huo & Tseng \(2017\)](#);

$$\rho^{t+1} = \begin{cases} \tau \rho^t & \text{if } \|r^t\|_2 > \eta \|s^t\|_2 \\ \rho^t / \tau & \text{if } \|r^t\|_2 < \eta \|s^t\|_2 \\ \rho^t & \text{otherwise} \end{cases},$$

where r^t is the primal residual and s^t is the dual residual at iteration t , defined above, and we set $\tau = 2$ and $\eta = 10$. This method helps to balance the primal and dual residuals so they converge to zero simultaneously.

2.2.4 Adaptive fused sparse group lasso

Zou (2006) showed that the lasso only exhibits consistent variable selection (i.e., identifies the right subset of non-zero coefficients asymptotically) under a certain nontrivial condition, which includes an orthogonal design matrix \mathbf{X} and $p = 2$ as special cases. The adaptive lasso, on the other hand, achieves consistent variable selection by differentially scaling the tuning parameter λ for each coefficient by the factor $|\hat{\beta}_j^*|^{-\gamma}$, where $\hat{\beta}_j^*$ is a consistent estimator for β_j such as the ordinary least squares estimator, and $\gamma > 0$ (Zou 2006). While the lasso biases nonzero coefficient estimates toward zero by a constant that is independent of coefficient magnitude, for the adaptive lasso the bias decreases as coefficients become large. Adaptive versions of fused lasso (Viallon et al. 2013) and group lasso (Wang & Leng 2008) have also been developed. In our application to a real neuroimaging dataset in Section 2.4, we implement an adaptive version of fused sparse group lasso using ridge regression to obtain initial coefficient estimates,

$$\hat{\boldsymbol{\beta}}^{\text{ridge}} = \arg \min_{\boldsymbol{\beta} \in \mathbb{R}^p} \frac{1}{2} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda^{\text{ridge}} \|\boldsymbol{\beta}\|_2^2.$$

The weights w_j introduced in Equation (2.5) are defined as

$$w_j = \begin{cases} \|\mathbf{j}_j \hat{\boldsymbol{\beta}}^{\text{ridge}}\|_1^{-1} & \text{if } j \in \{1, 2, \dots, p\} \\ \|\mathbf{d}_j \hat{\boldsymbol{\beta}}^{\text{ridge}}\|_1^{-1} & \text{if } j \in \{p+1, p+2, \dots, p+m\} \\ \|\mathbf{G}_j \hat{\boldsymbol{\beta}}^{\text{ridge}}\|_2^{-1} & \text{if } j \in \{p+m+1, p+m+2, \dots, p+m+G\}. \end{cases}$$

2.3 SIMULATION STUDY

2.3.1 Simulation study methods

This simulation study aimed to show that, for a given modeling scenario, the optimal weighting of the three penalty terms in the fused sparse group lasso depends upon the underlying structure of the true coefficients, and the study aimed to characterize the optimal penalty weights for a range of different coefficient structures. Accordingly, we evenly divided the

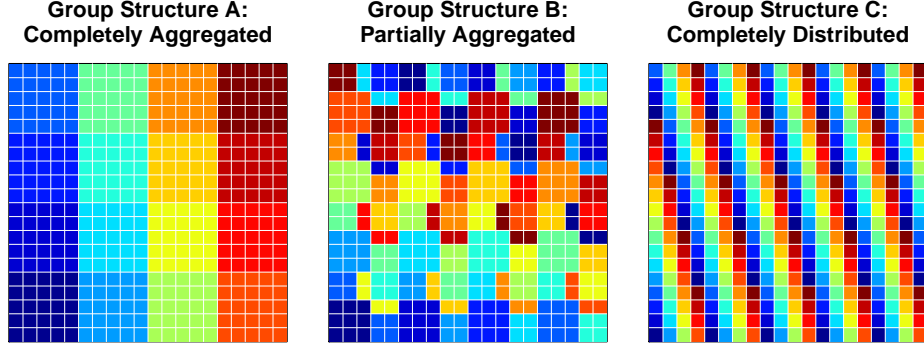


Figure 2.3: Simulation study group structures

pixels of two dimensional 20×20 images into 16 groups of 25 and considered three spatial arrangements of the groups (Figure 2.3): (A) members of a group were completely aggregated into 5×5 squares; (B) groups were partially aggregated, consisting of one 3×3 square, two 2×2 squares, and two 1×2 rectangles; (C) groups were completely distributed such that no pixels from the same group were touching sides. For each of these group structures, one group was selected to have non-zero coefficients, which were all set equal to 3. We also considered sparse versions of the coefficients, where 40% of coefficients in the active group were set to zero. Additionally, we considered three more scenarios under the partially aggregated group structure: an extra sparse scenario, with 80% of active group coefficients set to zero; a misspecified group structure, where the set of true coefficients was divided among several groups; and a sparse version of the misspecified group structure. Thus there were nine total scenarios of true coefficients (Figure 2.4).

For each of $n = 50$ training subjects and $n = 50$ test subjects, we generated a vector of 400 independent standard normal random variables to serve as predictors, where each corresponded to a pixel in the 20×20 image. The responses \mathbf{y} were then computed by the model $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$, where each element of $\boldsymbol{\epsilon}$ was independent normal with mean zero and variance 4. To select tuning parameters, we parameterized according to Equation (2.4). The fusion penalty was applied between coefficients of pixels that shared an edge, and the

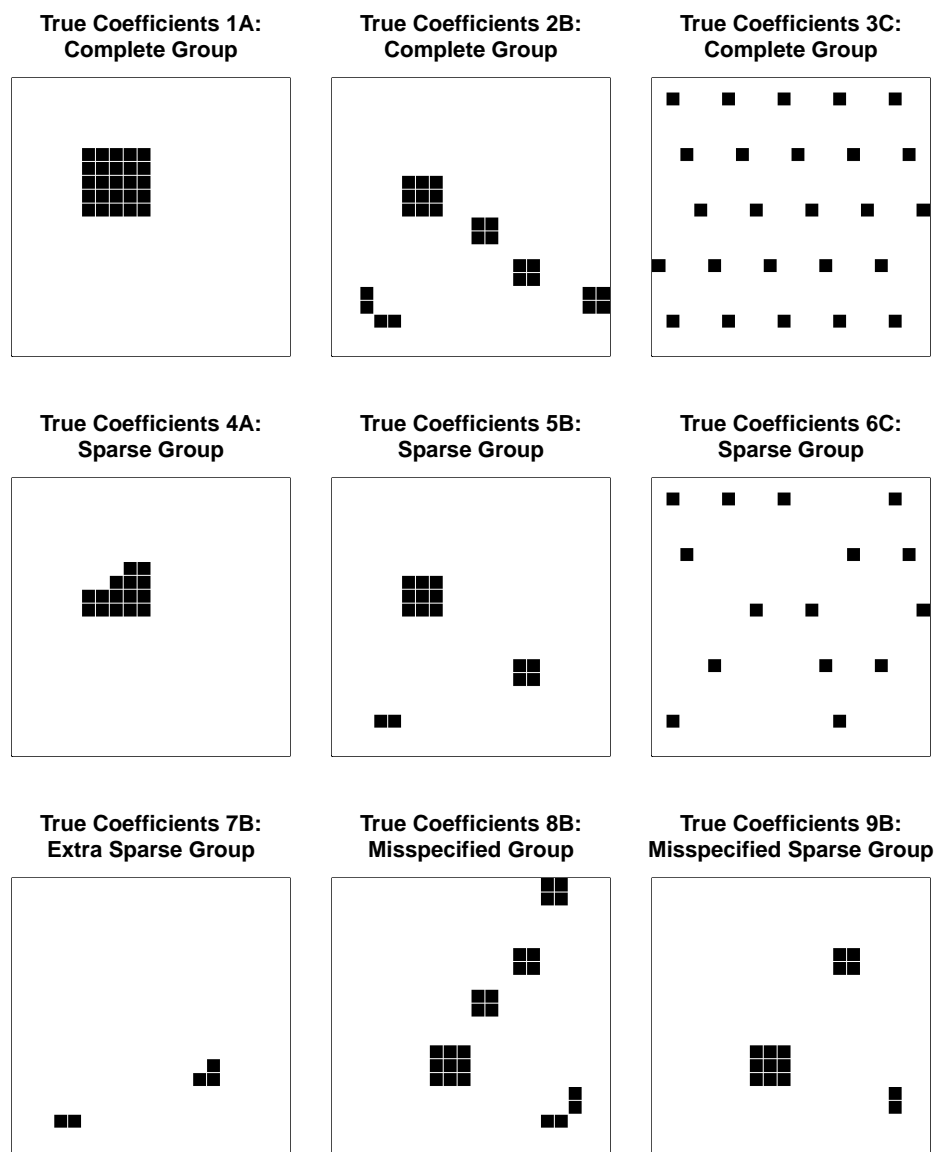


Figure 2.4: Simulation study true coefficients

group penalty was applied to each of the 16 groups as previously described. For each pair of $\alpha \in \{0, 0.2, 0.5, 0.8, 1\}$ and $\gamma \in \{0, 0.2, 0.5, 0.8, 1\}$, we performed 5-fold cross-validation over 50 values of $\lambda = 10^x$, where values of x formed a grid on the interval $[-3, 3]$, and selected the λ that resulted in the lowest cross-validation error. We fit the model to the entire sample of $n = 50$ training subjects at the given (α, γ) pair using this λ and calculated mean squared error of the estimated coefficients ($\text{MSE}(\hat{\beta}) = \|\hat{\beta} - \beta\|_2^2/400$) and mean squared prediction error for the test data ($\text{MSE}(\hat{\mathbf{y}}_{\text{test}}) = \|\hat{\mathbf{y}}_{\text{test}} - \mathbf{y}_{\text{test}}\|_2^2/50$). We repeated the entire procedure 100 times for each of the nine scenarios. We also generated another test sample of $n = 100$ for the purpose of decomposing the mean squared error into squared bias and variance at each (α, γ) combination across the 100 trained models. Tolerance levels were set to $\epsilon_{\text{abs}} = \epsilon_{\text{rel}} = 10^{-3}$, ADMM step-size was set to $\rho = 1$, and the maximum number of ADMM iterations was set to 2000. Analyses were done using R version 3.4.0 ([R Core Team 2017](#)).

We hypothesized that, on the basis of $\text{MSE}(\hat{\beta})$, $\text{MSE}(\hat{\mathbf{y}}_{\text{test}})$, or both, (1) the fusion penalty would perform worse and the sparsity penalties would perform better (i.e. optimal γ value would increase) as the groups became more spatially distributed; (2) the group penalty would perform worse and the ℓ_1 lasso penalty would perform better (i.e. optimal α value would increase) as the sparsity of true coefficients increased or with misspecification of group structure. We also sought to determine whether the lowest cross-validation error would correspond to the optimal values of (α, γ) .

2.3.2 Simulation study results

Optimal (α, γ) combinations for each scenario are presented in Table 2.2. Figures 2.5 – 2.7 show the distributions of cross-validation error, $\text{MSE}(\hat{\beta})$, and $\text{MSE}(\hat{\mathbf{y}}_{\text{test}})$ across the (α, γ) combinations for each of the nine scenarios. Decomposition of the mean squared error into squared bias and variance for each scenario is depicted in Figure 2.8. An example set of estimated coefficients for one simulation iteration of scenario 2B is shown in Figure 2.9. Detailed simulation results are tabulated in Appendix Tables A1 – A9.

Table 2.2: Optimal (α, γ) for each scenario

Optimal (α, γ) based on the most frequent lowest error out of 100 simulation iterations.

True coefficient scenario	Mean CVE	MSE($\hat{\beta}$)	MSE($\hat{\mathbf{y}}_{\text{test}}$)
1A. Completely aggregated	(0.0, 0.2)	(0.0, 0.2)	(0.0, 0.2)
2B. Partially aggregated	(0.0, 0.8)	(0.0, 0.8)	(0.0, 0.8)
3C. Completely distributed	(0.0, 1.0)	(0.0, 1.0)	(0.0, 1.0)
4A. Sparse completely aggregated	(0.0, 0.2)	(0.0, 0.2)	(0.0, 0.2)
5B. Sparse partially aggregated	(0.0, 0.5)	(0.0, 0.5)	(0.0, 0.5)
6C. Sparse completely distributed	(0.0, 1.0)	(0.0, 1.0)	(0.0, 1.0)
7B. Extra sparse partially aggregated	(0.8, 0.8)	(0.8, 0.8)	(0.8, 0.8)
8B. Misspecified partially aggregated	(1.0, 0.2)	(1.0, 0.2)	(1.0, 0.5)
9B. Misspecified sparse partially aggregated	(1.0, 0.2)	(1.0, 0.5)	(1.0, 0.5)

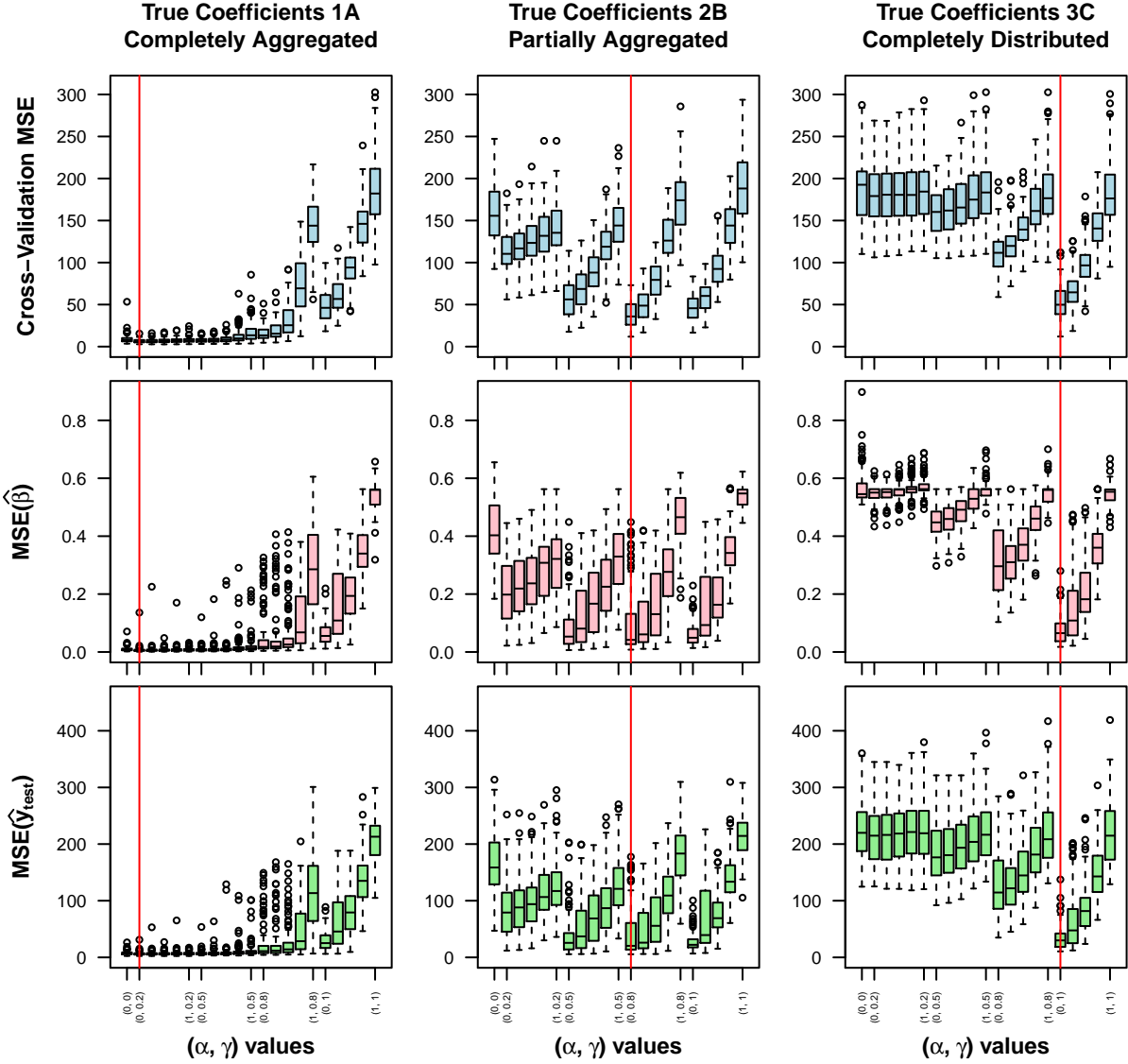


Figure 2.5: Simulation study results for true coefficients 1A, 2B, and 3C. Values of $\gamma \in \{0, 0.2, 0.5, 0.8, 1\}$ increase from left to right on the x -axis, corresponding to complete fusion penalty on the left ($\gamma = 0$) and complete sparsity penalties on the right ($\gamma = 1$). Intervals of increasing $\alpha \in \{0, 0.2, 0.5, 0.8, 1\}$ values correspond to complete group penalty on the left ($\alpha = 0$) and complete ℓ_1 lasso penalty on the right ($\alpha = 1$). Red vertical lines indicate (α, γ) combination yielding most frequent lowest error over 100 simulations. MSE: mean squared error.

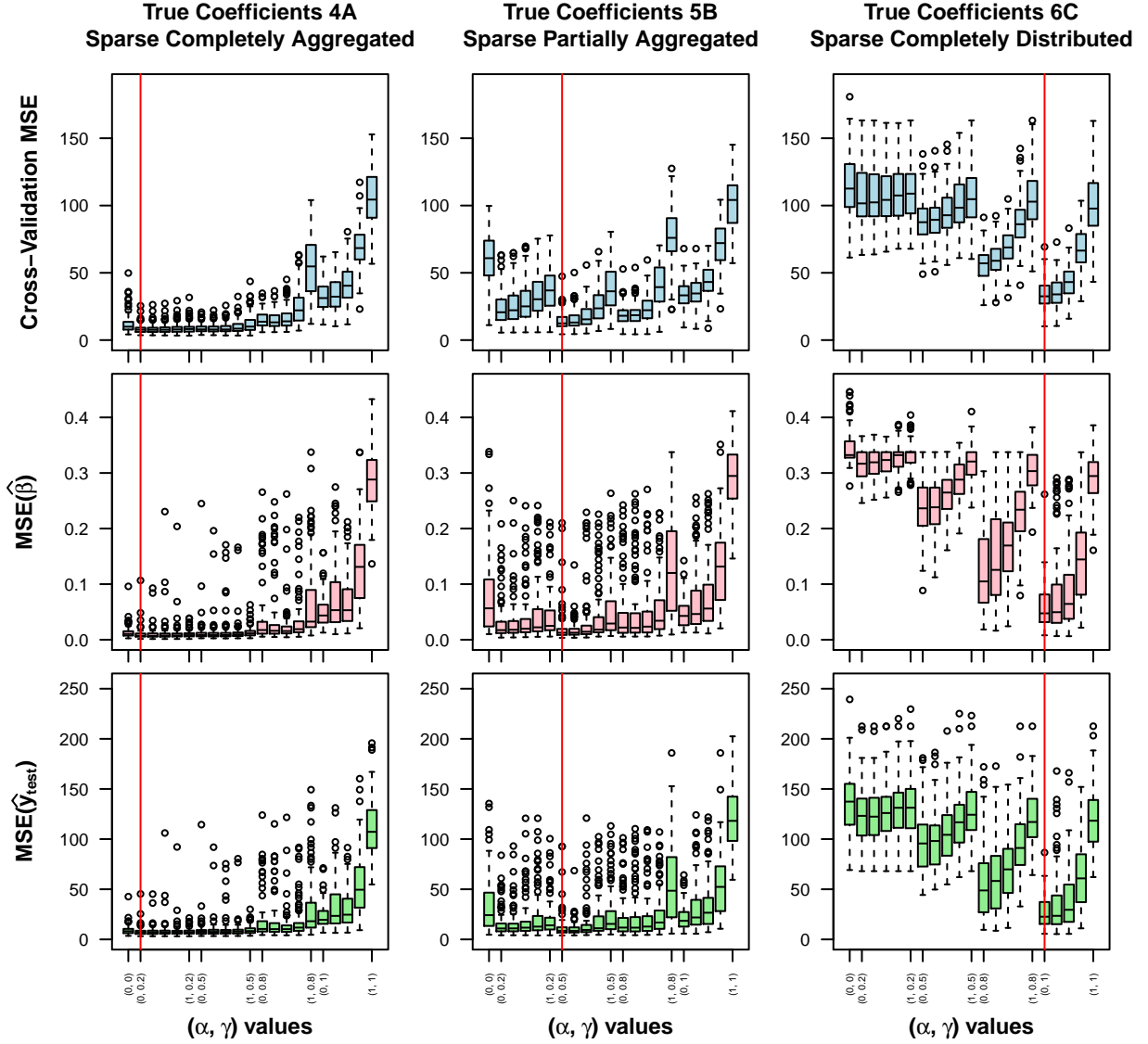


Figure 2.6: Simulation study results for true coefficients 4A, 5B, and 6C. Values of $\gamma \in \{0, 0.2, 0.5, 0.8, 1\}$ increase from left to right on the x -axis, corresponding to complete fusion penalty on the left ($\gamma = 0$) and complete sparsity penalties on the right ($\gamma = 1$). Intervals of increasing $\alpha \in \{0, 0.2, 0.5, 0.8, 1\}$ values correspond to complete group penalty on the left ($\alpha = 0$) and complete ℓ_1 lasso penalty on the right ($\alpha = 1$). Red vertical lines indicate (α, γ) combination yielding most frequent lowest error over 100 simulations. MSE: mean squared error.

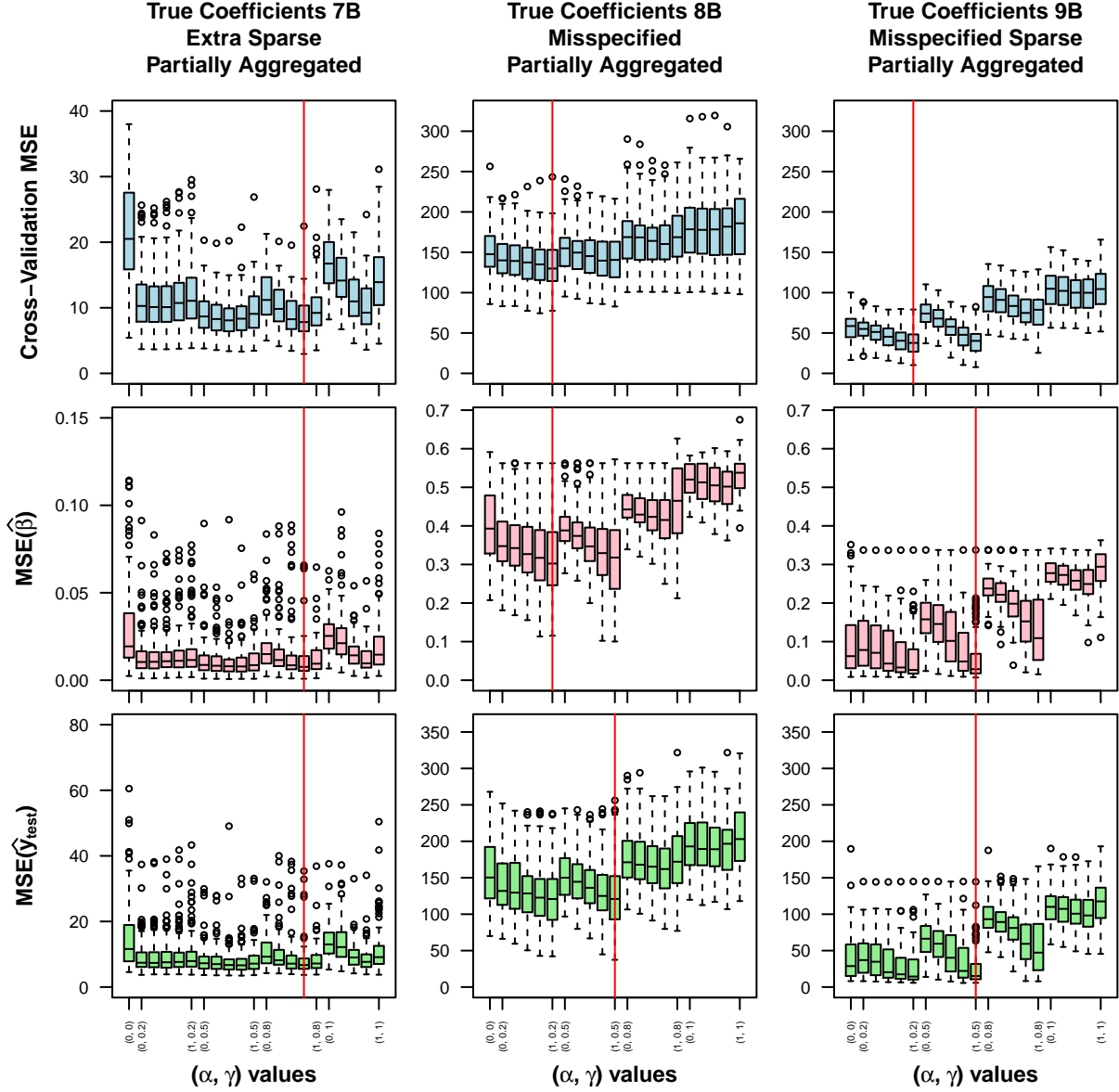


Figure 2.7: Simulation study results for true coefficients 7B, 8B, and 9B. Values of $\gamma \in \{0, 0.2, 0.5, 0.8, 1\}$ increase from left to right on the x -axis, corresponding to complete fusion penalty on the left ($\gamma = 0$) and complete sparsity penalties on the right ($\gamma = 1$). Intervals of increasing $\alpha \in \{0, 0.2, 0.5, 0.8, 1\}$ values correspond to complete group penalty on the left ($\alpha = 0$) and complete ℓ_1 lasso penalty on the right ($\alpha = 1$). Red vertical lines indicate (α, γ) combination yielding most frequent lowest error over 100 simulations. MSE: mean squared error.

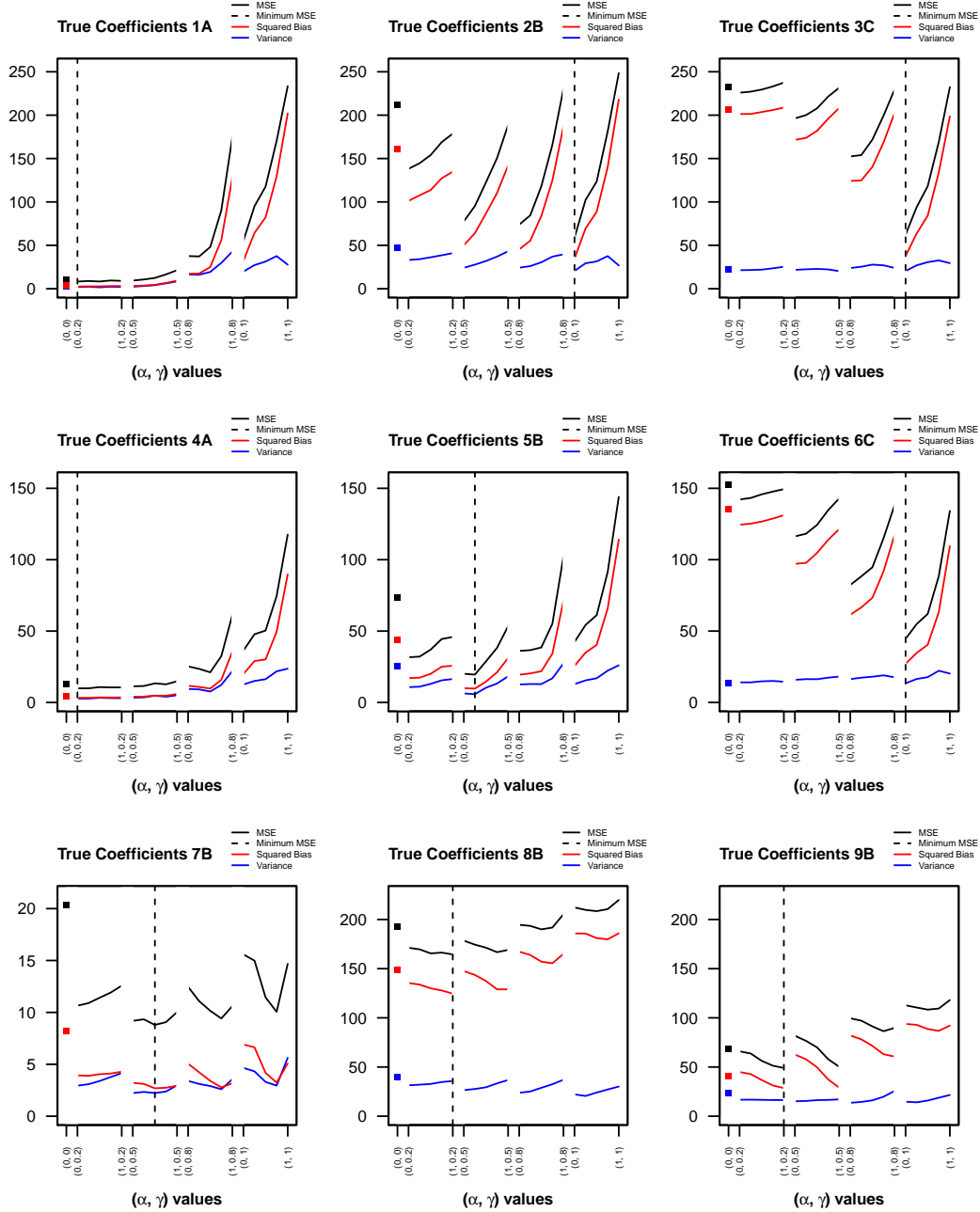


Figure 2.8: Simulation study squared bias, variance, and MSE of predicted responses. Mean over $n = 100$ test observations of the squared bias, variance, and mean squared error of predicted responses from models estimated in 100 simulation iterations. Values of γ increase from left to right on the x -axis, corresponding to complete fusion penalty on the left ($\gamma = 0$) and complete sparsity penalties on the right ($\gamma = 1$). Intervals of increasing α values correspond to complete group penalty on the left ($\alpha = 0$) and complete ℓ_1 lasso penalty on the right ($\alpha = 1$).

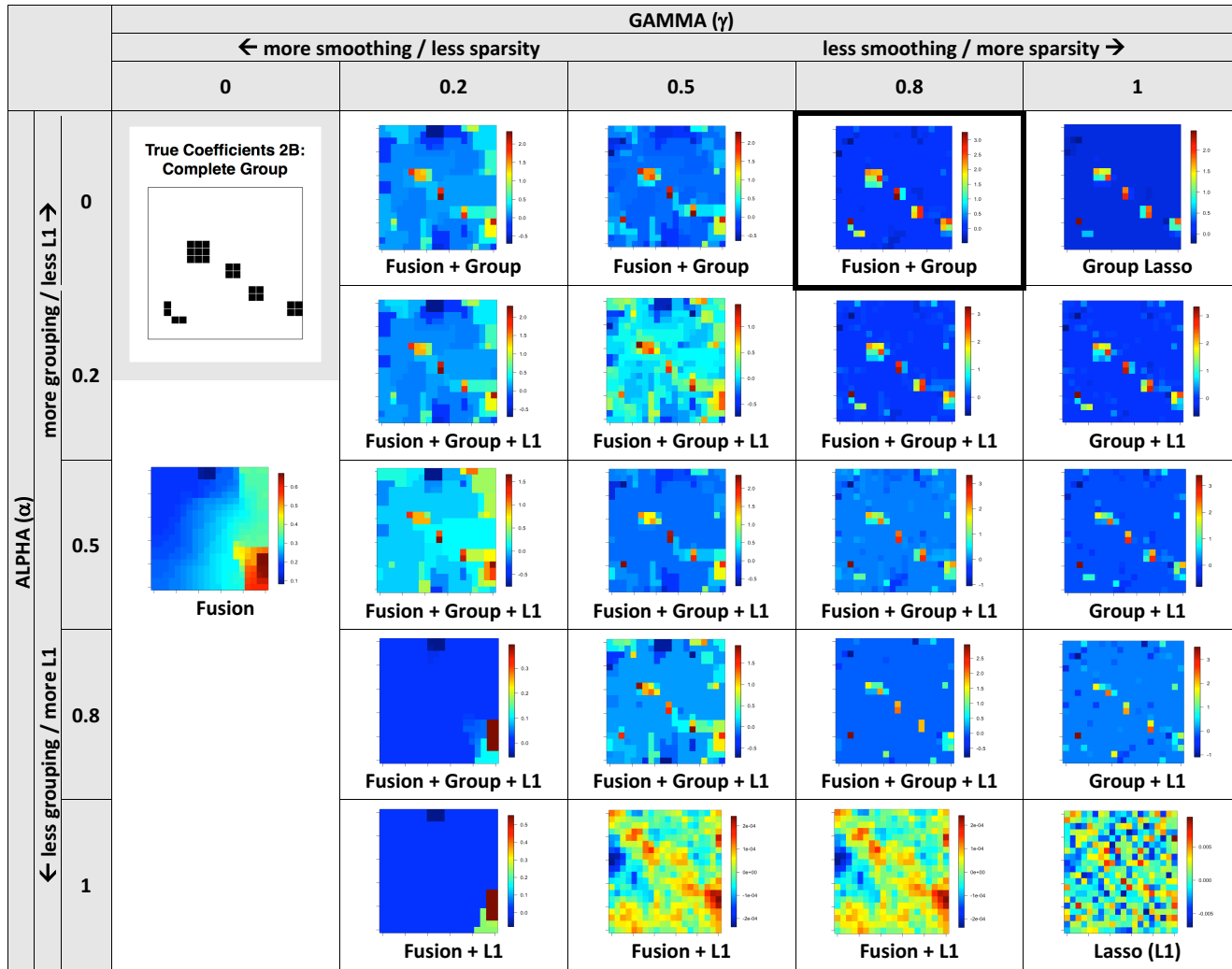


Figure 2.9: Example set of estimated coefficients for one simulation iteration of scenario 2B. For this scenario, best performance was observed when $\alpha = 0$ and $\gamma = 0.8$ (noted as cell with thicker border). Note: Color scales vary across cells.

As expected, on the basis of both $\text{MSE}(\hat{\beta})$ and $\text{MSE}(\hat{\mathbf{y}}_{\text{test}})$, as groups became more spatially distributed, the optimal value of γ increased from 0.2 for the completely aggregated to 1 for the completely distributed group structure, shifting weight from the fusion penalty term to the sparsity penalty terms. This pattern was similar for the complete (1A, 2B, 3C) and sparse (4A, 5B, 6C) group scenarios. As the sparsity of true coefficients increased in the partially aggregated group scenarios (2B, 5B, 7B), the optimal value of α increased from 0 for the complete group to 0.8 for the extra sparse group scenario, shifting weight from the group penalty term to the ℓ_1 lasso penalty term. When group structure was misspecified, the optimal α value was 1 for both scenarios (8B, 9B), putting zero weight on the group penalty term in favor of the ℓ_1 lasso penalty term.

The results demonstrate that the combination of penalty terms in the fused sparse group lasso adapt to a wide range of spatial arrangements and sparsity levels of true coefficients. When group structure was correctly specified, a combination of group and fusion penalty terms was optimal for scenarios including spatially aggregated groups of true coefficients (1A, 2B, 4A, 5B), while the group penalty term alone was optimal for the completely distributed true coefficient scenarios (3C, 6C). When group structure was misspecified (8B, 9B), the fusion and ℓ_1 penalty terms together were optimal, and for the extra sparse partially aggregated scenario (7B), all three penalty terms had non-zero weights. In all seven scenarios where group structure was correctly specified, the (α, γ) combination yielding the most frequent lowest cross-validation error corresponded to the most frequent lowest $\text{MSE}(\hat{\beta})$ and $\text{MSE}(\hat{\mathbf{y}}_{\text{test}})$, indicating that selecting tuning parameters based on lowest cross-validation error tends to correspond to the optimal model. For the misspecified group scenarios, cross-validation error was lowest for either the first or second most frequent lowest $\text{MSE}(\hat{\beta})$ and $\text{MSE}(\hat{\mathbf{y}}_{\text{test}})$ (see Appendix Tables A8 and A9).

2.4 APPLICATION TO ABIDE NEUROIMAGING DATA

Autism spectrum disorder (ASD) is a group of developmental disorders characterized by impaired social functioning and restrictive and repetitive behavior, and affects approximately

1% of children (Baio 2012). Neuroimaging studies report abnormal functional connectivity between brain regions in ASD, although findings are mixed regarding the specific nature of the abnormalities (Di Martino et al. 2014). In the following application, we show that incorporating prior information about voxel spatial location and functional connectivity using fused sparse group lasso increases accuracy and produces smoother estimated coefficient maps when predicting a continuous measure of autistic social impairment, Social Responsiveness Scale (SRS, Constantino et al. (2003)) score, from resting state fMRI data.

We applied fused sparse group lasso (FSGL) penalized regression to a resting state fMRI dataset of autism spectrum disorder ($n = 111$) and typically developing (TD, $n = 108$) male participants (mean (SD) age 17.4 (7.5) years, see Table B1 for descriptive summary) from the Autism Brain Imaging Data Exchange (ABIDE) repository (Di Martino et al. (2014), http://fcon_1000.projects.nitrc.org/indi/abide/). In this set of participants, Cerliani et al. (2015) used independent components analysis to identify 19 resting state brain networks. The authors found that autistic social impairment, as measured by SRS score, was positively associated with functional connectivity between a subcortical network, comprising basal ganglia and thalamus, and two cortical networks: (1) dorsal and (2) ventral primary somatosensory and motor cortices. The association was only significant in the ASD group. Given that the resting state networks evaluated in Cerliani et al. (2015) represent relatively large brain regions, we used FSGL regression to more precisely define the cortical regions whose functional connectivity with a subcortical seed region best predicts SRS scores.

2.4.1 Application methods

Preprocessed fMRI data was downloaded from the ABIDE I Preprocessed repository (Craddock et al. 2013). Data were preprocessed using the Connectome Computational System pipeline with no global signal regression and band pass filtering (0.01 – 0.1 Hz) (details at <http://preprocessed-connectomes-project.org/abide/Pipelines.html>). The independent component resting state network data from Cerliani et al. (2015) was downloaded from <https://github.com/sblnin/rsfnc> and resampled to $3 \times 3 \times 3$ mm³ voxels to match the ABIDE data.

We partitioned the brain into 19 resting state networks by assigning each voxel to the maximal spatial independent component at that voxel out of the 19 components identified in [Cerliani et al. \(2015\)](#). We restricted our analyses to the three networks mentioned above: the basal ganglia/thalamus subcortical network and the two sensorimotor cortical networks. See [Figure 2.10](#) for an overview of the following methods. We defined a subcortical seed region by selecting the peak voxels in the subcortical network independent component spatial map. This yielded bilateral regions of the thalamus, with 12 voxels centered at MNI coordinates $(11, -11, 11)$ and 11 voxels centered at $(-11, -11, 12)$ ([Figure 2.11A](#)). For each participant, the first eigenvariate of the seed region time series was extracted, and its Pearson correlation was calculated with each voxel time series in the cortical regions of interest to form a seed-based connectivity map. Fisher’s r -to- z transformation was applied to each voxel. After excluding voxels where any participants had missing data, this left $p = 5476$ voxels to serve as predictors for FSGL regression ([Figure 2.11B](#)).

Participant data were divided into training ($n = 175$) and test ($n = 44$) sets. Test set data was put aside until all model fitting was completed. The following steps were carried out with the training set data:

- (1) A linear regression model adjusted raw SRS scores for age, full-scale IQ, site of acquisition, eye status at scan (open or closed), and mean framewise displacement. The residuals were used as the outcome for FSGL regression. (See [Appendix Table B2](#).)
- (2) The fusion penalty was applied between coefficients of voxels who shared a face, so each voxel had a maximum of 6 neighbors. For the group penalty term, voxels in the cortical regions of interest were partitioned into 50 groups using agglomerative hierarchical clustering. First, for each training participant, Pearson correlation between time series was calculated for all possible pairs of voxels in the cortical regions of interest. Correlation matrices were averaged across participants, and a distance matrix was formed by applying the element-wise transformation $d = \sqrt{2 * (1 - r)}$. Finally, hierarchical clustering using Ward’s method was performed based on the distance matrix, and the resulting tree was cut to form 50 groups which ranged in size from 43 to 268 voxels ([Figure 2.11C](#)).
- (3) After centering the SRS outcome and standardizing columns of the predictor matrix, 5-fold cross-validation was used to determine the λ value yielding the minimum cross-

validation error at selected values of (α, γ) for the FSGL regression. We chose to compare (α, γ) equal to (1.0, 1.0) (standard lasso), (0.2, 1.0) (sparse group lasso), (0.2, 0.8) (fused sparse group lasso), and (0.0, 0.8) (fused group lasso). Cross-validation folds were stratified to ensure that they had similar distributions of the adjusted SRS outcome.

- (4) To estimate the coefficients, the model was fit to the entire training set at the optimal λ for selected values of (α, γ) .
- (5) For adaptive FSGL regression, first ridge regression estimates were obtained (using R package `glmnet`, [Friedman et al. \(2010\)](#)) at the optimal λ^{ridge} determined by 5-fold cross-validation, adaptive weights were formed according to Equation 2.2.4, and then steps (3) and (4) were completed.

The ADMM tolerance levels were set to $\epsilon_{\text{abs}} = \epsilon_{\text{rel}} = 10^{-3}$, step-size was set to $\rho = 1000$, and the maximum number of iterations was set to 6000.

For the test set data, predictor variables were first standardized according to the training set column means and standard deviations, and then predicted adjusted SRS scores were obtained using the estimated coefficients. Raw test set SRS scores were adjusted using the linear regression model parameters estimated with the training set data. Prediction accuracy was assessed via mean squared error and Pearson correlation of predicted with actual adjusted SRS scores. Since prior studies have used resting state fMRI data to classify ASD and TD subjects into diagnostic groups rather than predict SRS, in order to compare our results we used receiver operating characteristic (ROC) curve analysis to find the best classification threshold for predicted SRS scores from the best-performing models and calculated the corresponding classification accuracies.

Analyses were done using a combination of AFNI version 17.1.03 ([Cox 1996](#)), MATLAB version 9.1.0 (R2016b) ([MATLAB 2016](#)) with SPM12 software ([Ashburner et al. 2014](#)), and R version 3.4.0 ([R Core Team 2017](#)).

2.4.2 Application results

Results for non-adaptive and adaptive fused sparse group lasso as well as for ridge and elastic net penalties are summarized in Table 2.3. The best test set prediction was achieved

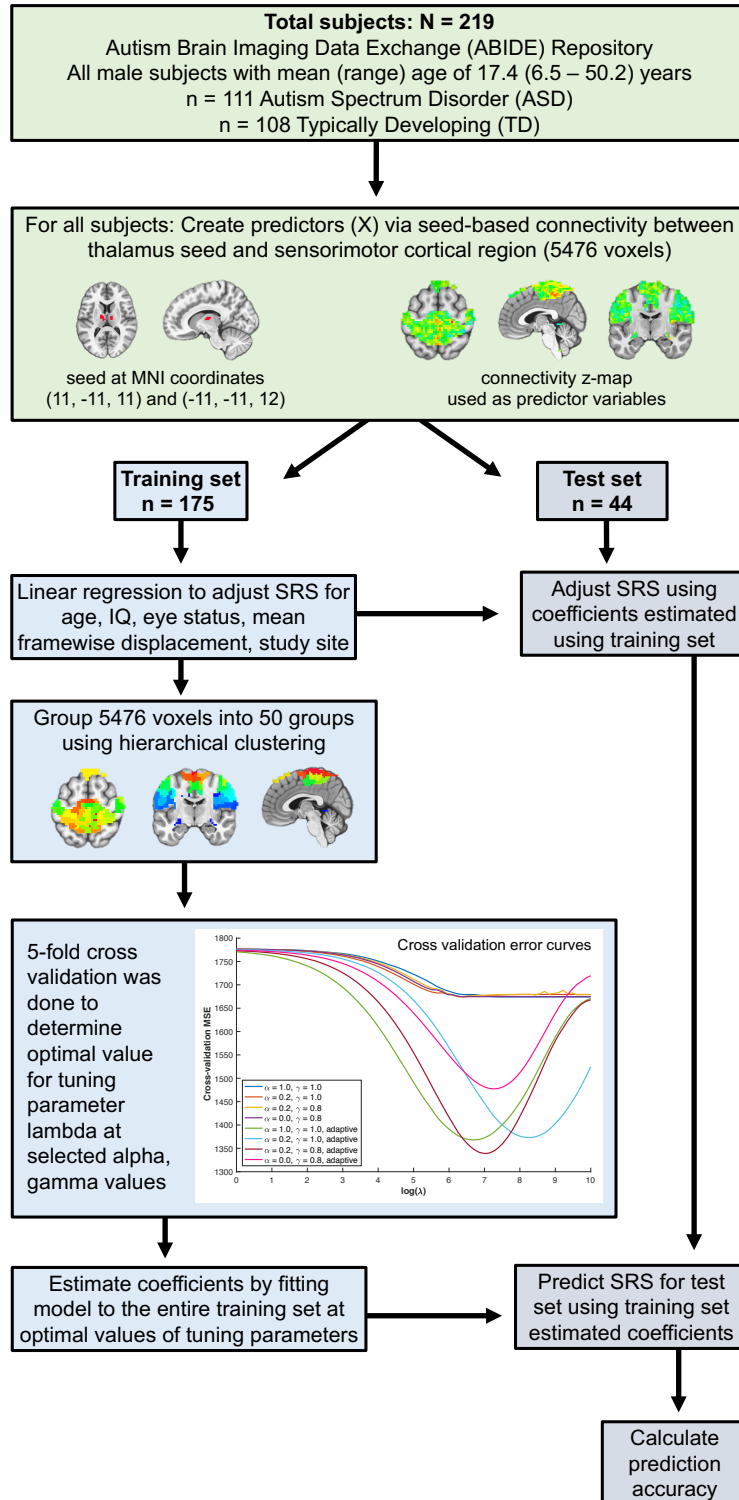


Figure 2.10: Overview of ABIDE application methods. SRS: Social Responsiveness Scale

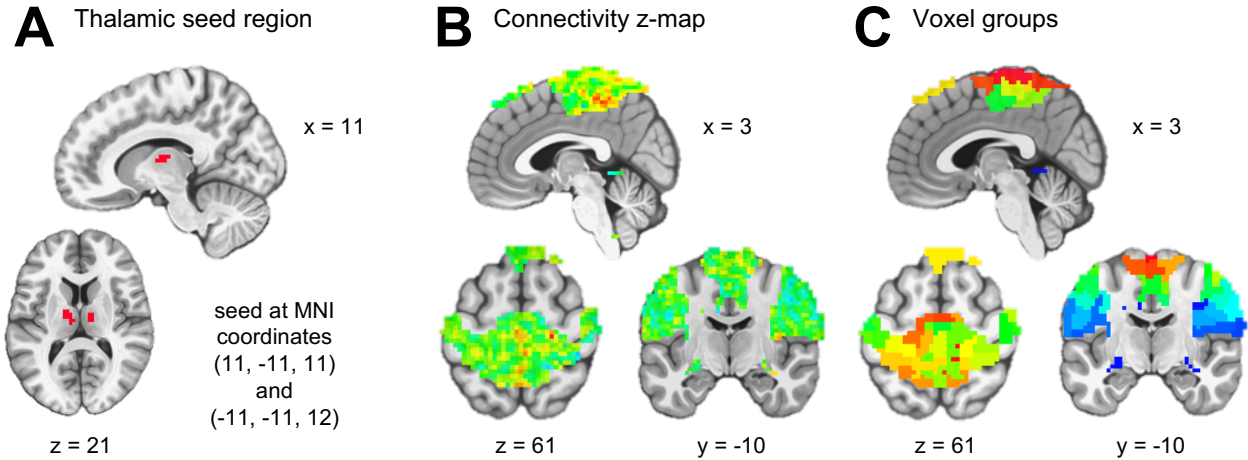


Figure 2.11: Fused sparse group lasso regression applied to ABIDE dataset, methods. (A) The thalamic seed region consisted of 23 $3 \times 3 \times 3$ mm³ voxels. (B) Connectivity z-maps of 5476 voxels for each participant served as predictors for fused sparse group lasso regression. (C) Voxels were partitioned into 50 groups using agglomerative hierarchical clustering on the training set resting state fMRI voxel time series. MNI: Montreal Neurological Institute.

by the adaptive fused sparse group lasso with (α, γ) equal to (0.2, 0.8), which gave mean squared error of 1165.2 and Pearson correlation $r = 0.437$ ($p = 0.003$) (Figure 2.12B). (For comparison, Cerliani et al. (2015) reported correlations of $r = 0.21$ and 0.25 between subcortical-cortical functional connectivity and SRS score for the respective cortical networks in ASD participants.) Cross-validation error was in general much lower for adaptive penalties than for non-adaptive penalties (Figure 2.12A). Adaptive penalties likely show better cross-validation performance in part because the ridge regression coefficients used for weighting were estimated on the entire training set, thus introducing a downward bias into the cross-validation error. To avoid this bias, ridge regression coefficients could be estimated separately for each cross-validation fold. This does not explain better performance of adaptive penalties on the independent test set, however. The superior performance of adaptive, ridge, and elastic net penalties over the non-adaptive penalties in this particular application could be due to high multicollinearity of the predictors and the influence of many small, weak effects of predictors on the outcome rather than a few strong effects.

Estimated coefficient brain maps are shown in Figure 2.12C. Higher SRS scores correspond to greater autistic social impairment. Thus the coefficient maps reflect multivariate thalamic seed connectivity patterns predictive of greater social impairment. Penalties including the fusion term, i.e., with $\gamma = 0.8$, resulted in larger clusters of contiguous regions, rather than the more scattered coefficient maps resulting when $\gamma = 1.0$. Figure 2.13 shows estimated FSGL coefficients plotted against the ridge regression estimates, which provides insight into how differential weighting of the penalty terms affects the structure of estimated coefficients.

A couple of questions arise regarding the scientific and clinical significance of the findings. First, does the result provide insight into the neurobiology of ASD? While the sparse, structured penalty succeeded in narrowing down the predictors to a smaller subset of the most predictive voxels, it is not immediately clear why these particular scattered regions of sensorimotor cortex are most informative. We invite interested readers to further explore the coefficient brain maps available at <https://github.com/jcbeer/fs-gl>. Second, does the result represent a good diagnostic biomarker? We consider this question in the following context. ASD has been associated with abnormalities in connectivity between multiple brain regions (Di Martino et al. 2014). Accordingly, studies using resting state fMRI data to define ASD biomarkers have often summarized voxel-wise data using regions of interest and considered connectivity between multiple regions, rather than use a focused, voxel-wise approach as we did. Previous studies using such methods on the ABIDE dataset have achieved diagnostic classification (ASD versus TD) accuracies in the range of 60% to 71% (Abraham et al. 2017, Nielsen et al. 2013, Plitt et al. 2015, Kassraian-Fard et al. 2016), well below the classification accuracy that can be achieved using behavioral measures such as the SRS, which can attain accuracies of up to 95% (Plitt et al. 2015). The difficulty of identifying fMRI biomarkers for ASD may be due to the noisiness of fMRI data or the neurobiological heterogeneity of the disorder (Plitt et al. 2015). What is remarkable about our result is that, when we dichotomize our predicted outcomes for the purpose of diagnostic classification, we can achieve similar accuracies (classification accuracy of 29 to 30 out of 44 test subjects, or 66% to 68%, for the adaptive penalties), even though we only considered connectivity between a single thalamic seed region and sensorimotor cortex. This may reflect the rich-

ness imparted by voxel-wise as opposed to region of interest functional connectivity data. It seems possible that adding other features to the model, e.g., using not only a thalamic seed region, but also including voxel-wise connectivity data from other seed regions that have shown abnormal connectivity in ASD, such as the default mode network and regions implicated in social cognition, would likely improve prediction performance further.

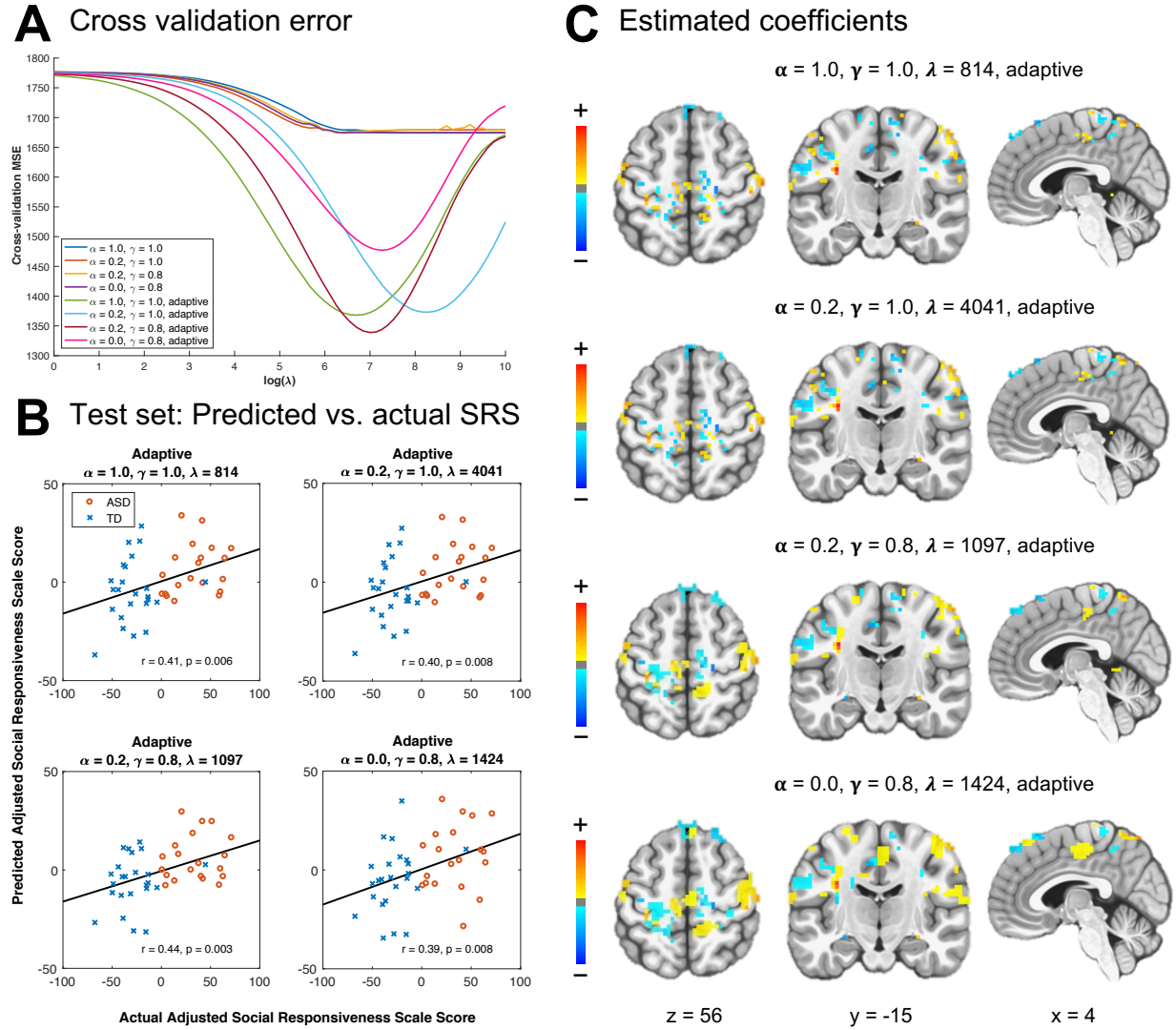


Figure 2.12: Fused sparse group lasso regression applied to ABIDE dataset, results. (A) Five-fold cross-validation was carried out over a range of λ values for several sets of α and γ values. (B) Correlation of predicted and actual adjusted SRS scores for selected α and γ values. Points are distinguished by autism spectrum disorder (ASD, red circle) and typically developing (TD, blue cross) diagnosis groups. (C) Estimated coefficients at the optimal λ for selected α and γ values. Higher coefficient values contribute to higher predicted Social Responsiveness Scale (SRS) scores, which indicate greater autistic social impairment.

Table 2.3: Comparison of estimators applied to ABIDE dataset

Method	α	γ	Estimator	* Optimal λ	Training set ($n = 175$)				Test set ($n = 44$)		
					CVMSE	MSE	r	p	MSE	r	p
Glmnet	0.0		Ridge	2627	1646.7	954.3	0.879	< 0.001	1325.8	0.285	0.060
	0.01		Elastic Net	289	1661.2	935.7	0.883	< 0.001	1305.4	0.320	0.034
FSGL	1.0	1.0	Lasso	1848	1674.2	1689.1	0.159	0.036	1426.7	0.035	0.821
	0.2	1.0	Sparse Group Lasso	521	1674.4	1572.0	0.383	< 0.001	1427.8	0.069	0.654
	0.2	0.8	Fused Sparse Group Lasso	604	1673.9	1633.2	0.254	< 0.001	1434.3	0.038	0.805
	0.0	0.8	Fused Group Lasso	604	1674.3	1641.2	0.232	0.002	1435.4	0.032	0.838
Adaptive FSGL	1.0	1.0	Adaptive Lasso	814	1368.1	120.1	0.986	< 0.001	1193.1	0.406	0.006
	0.2	1.0	Adaptive Sparse Group Lasso	4041	1373.2	129.1	0.985	< 0.001	1203.1	0.397	0.008
	0.2	0.8	Adaptive Fused Sparse Group Lasso	1097	1338.9	168.6	0.977	< 0.001	1165.2	0.437	0.003
	0.0	0.8	Adaptive Fused Group Lasso	1424	1477.2	144.2	0.981	< 0.001	1211.6	0.394	0.008

* Note: λ for glmnet R package is scaled by factor n^{-1}

Mean total sum of squares for training set = 1697.5

Mean total sum of squares for test set = 1428.0

ABIDE: Autism Brain Imaging Data Exchange

FSGL: fused sparse group lasso

CVMSE: cross-validation mean squared error

MSE: mean squared error

r : Pearson correlation

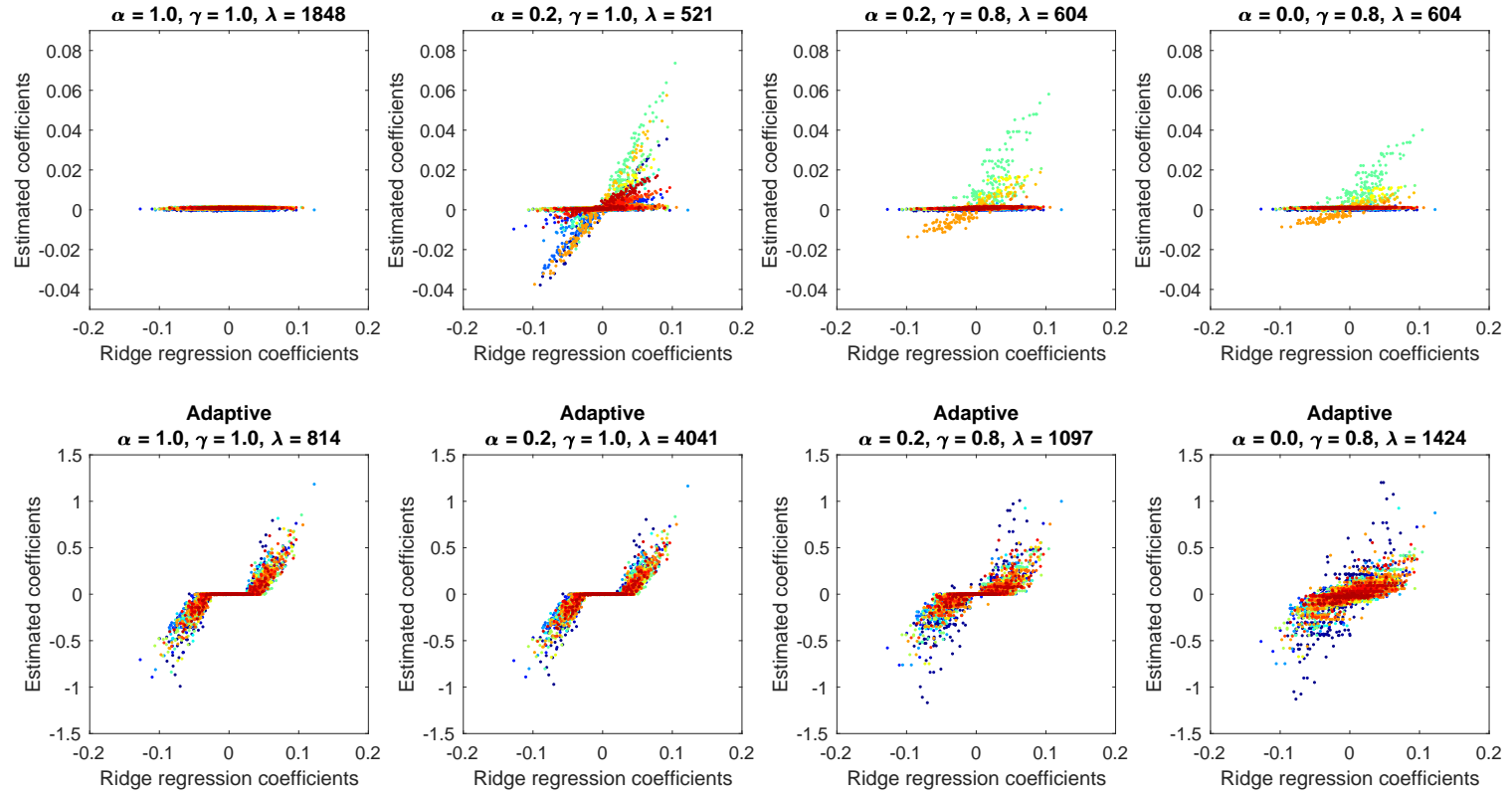


Figure 2.13: ABIDE dataset estimated coefficients. For various fused sparse group lasso estimators, plotted against the ridge regression estimated coefficients. Plotting points are colored by voxel group membership.

2.5 CONCLUSIONS

The fused sparse group lasso penalty offers a flexible way to incorporate prior information into a predictive model, which can lead to more interpretable coefficient estimates and better prediction performance on test data. The fusion penalty term constrains coefficients that we expect to have similar estimated values, and we can use it to enforce local spatial smoothness in an image. The group penalty term groups together coefficients that we do not necessarily expect to have similar values, but we expect to be selected simultaneously, such as voxels residing in the same functional brain networks. The ℓ_1 penalty term allows sparse groups, and may also be useful when groups are misspecified. Cross-validation over a range of weights for the three penalty terms allows a data-driven way of incorporating information about coefficient structure into a prediction model.

In this chapter we have presented an ADMM optimization algorithm to fit fused sparse group lasso. A simulation study featuring a range of coefficient structures demonstrated instances where a combination of the three penalty terms together outperforms any smaller subset, and showed that cross-validation is a reliable way to select optimal tuning parameter weights. On real fMRI data, we found that incorporating adaptive weights derived from initial ridge regression coefficient estimates greatly improved performance over non-adaptive fused sparse group lasso as well as ridge and elastic net penalties. The adaptive fused sparse group lasso produced the best test set prediction, and the addition of fusion and group penalty terms resulted in less dispersed, more clustered coefficient maps. Fused sparse group lasso, a generalization of lasso, group lasso, and fused lasso, has potential application not only to prediction problems in neuroimaging, but also to other contexts where coefficients are expected to be both smooth and group-structured.

3.0 FUSED SPARSE GROUP LASSO FOR MULTIMODAL NEUROIMAGING PREDICTION

3.1 INTRODUCTION

Different neuroimaging modalities, such as magnetic resonance imaging (MRI), positron emission tomography (PET), electroencephalography (EEG), and magnetoencephalography (MEG), provide different information about brain structure and function. Modalities differ in their spatial and temporal resolution as well as the physical and biological properties of the brain they measure. The practice known as multimodal data fusion involves combining this complementary information together to make inferences or predictions.

Several studies have demonstrated the benefits of using multimodal data over single modalities alone for the purposes of classification or prediction, and reviews of multimodal neuroimaging report an exponentially increasing number of multimodal studies ([Liu et al. 2015](#), [Calhoun & Sui 2016](#)). Moreover, [Liu et al. \(2015\)](#) report increasing use of multimodal neuroimaging in the assessment of neuropsychiatric disorders, informing diagnosis, prognosis, and treatment decisions. These trends present a need for computational methods and models that can handle the complexity and high dimensionality of multimodal neuroimaging data and extract clinically useful information such as neuroimaging biomarkers.

In this chapter, we extend fused sparse group lasso to the multimodal setting. We show how fused sparse group lasso can be used to predict age from multimodal neuroimaging data and yield an interpretable neuroimaging biomarker for age. In [Section 3.1.1](#) we review age-related structural and functional changes in the brain, and in [Section 3.1.2](#) we describe how predicting age from neuroimages can be clinically useful and discuss previous work in this area. In [Section 3.2](#) we outline a multimodal version of fused sparse group lasso, and in

Section 3.3 we describe an application of fused sparse group lasso to multimodal neuroimaging data from a normal aging study. Finally, we present discussion and conclusions in Section 3.4.

3.1.1 Age-related structural and functional changes in the brain

Macroscopic age-related structural changes in the brain include ventricular enlargement, cortical thinning, decreased total brain volume, and white matter hyperintensities on structural MRI scans, which can reflect changes due to small vessel disease or other types of tissue damage such as demyelination (Cole et al. 2018, Wardlaw et al. 2015). Regarding particular spatial patterns of brain atrophy seen in normal aging, results differ across studies depending on the methods used (e.g., manually defined region of interest approaches versus more recent automated voxel-based approaches) and study design (e.g., cross-sectional versus longitudinal studies). (For a review, see Fjell & Walhovd (2010).) However, in general, the largest changes appear to be in the frontal and temporal cortices, putamen, thalamus, and nucleus accumbens (Fjell & Walhovd 2010). Longitudinal studies have reported atrophy rates greater than those reported in cross-sectional studies for particular regions such as the hippocampus and entorhinal cortex, and additionally report accelerating atrophy with age in particular regions, including hippocampus, entorhinal cortex, inferior temporal cortex, prefrontal white matter, and cerebellum (Raz et al. 2005, Fjell & Walhovd 2010).

Functional changes in the normally aging brain have also been noted, many of them in the default mode network (DMN, Raichle et al. (2001)). For example, older age has been associated with reduced deactivation of the DMN during semantic classification (Lustig et al. 2003) and memory tasks (Grady et al. 2006). Damoiseaux et al. (2007) also found decreased resting state DMN activity in healthy older versus younger adults. Andrews-Hanna et al. (2007) examined the functional integrity of the DMN during a semantic classification task in younger and older adults and found the largest age-related changes in the disruption of anterior to posterior connectivity of the DMN. These changes were associated with white matter degradation, as measured by diffusion tensor imaging, and poor cognitive performance, but not brain amyloid beta status. Tomasi & Volkow (2012) report age-related resting state

functional connectivity decreases in DMN as well as regions of the dorsal attention network, and functional connectivity increases in sensorimotor cortex, amygdala, and thalamus. The authors found that functional connectivity decreases were more pronounced for long-range connections, which they speculate may be due to deterioration of axonal myelin sheaths with age or degeneration associated with the larger distances proteins must travel in longer axons.

3.1.2 Predicting age from neuroimages

Recently, increasing attention has focused on predicting age from neuroimages, most commonly structural MRI, using various machine learning methods (for a review, see [Cole et al. \(2018\)](#)). When a model is trained on a large set of healthy subjects, and the model is then used to predict brain age in clinical samples, the difference between chronological and predicted age can indicate whether individuals deviate from a pattern of healthy brain aging. Here we follow the convention in [Cole et al. \(2018\)](#) and adopt the term *brain-predicted age difference* (brain-PAD) to refer to this difference, i.e., predicted age minus chronological age. Many neuropsychiatric disorders and diseases have been associated with a brain phenotype consistent with accelerated aging, including Down’s syndrome, schizophrenia, depression, epilepsy, traumatic brain injury, mild cognitive impairment, and Alzheimer’s disease (AD) ([Cole et al. 2018](#)). A positive brain-PAD has been associated with cognitive decline and poor clinical outcomes. For example, increased brain-PAD was found in adults with mild cognitive impairment (MCI) who progress to AD ([Franke & Gaser 2012](#), [Gaser et al. 2013](#)). Greater brain-PAD has also been associated with less education and less self-reported exercise ([Steffener et al. 2016](#)), and increased risk of mortality and poor performance on other age-associated functional measures ([Cole, Ritchie, Bastin, Hernández, Maniega, Royle, Corley, Pattie, Harris, Zhang et al. 2017](#)). Brain age predicted from neuroimaging data can potentially serve as an important biomarker for preclinical AD since changes in the brain tend to occur before clinical symptoms such as cognitive decline ([Jack Jr et al. 2013](#)). Below, we briefly review several studies that have successfully predicted brain age from neuroimages with high accuracy using machine learning methods.

Franke et al. (2010) predicted age using T1-weighted structural MRI data (preprocessed to produce grey matter density maps) and relevance vector regression in a sample of healthy adults aged 19 to 86. They used a training set of $n = 410$ and a test set of $n = 245$. For the test set, they achieved a correlation between predicted and chronological age of $r = 0.92$ and a mean absolute error (MAE) of 5 years. When they then applied the trained age prediction model to $n = 102$ participants with mild AD (i.e., with a global Clinical Dementia Rating (CDR, Morris (1993)) of 1) and $n = 232$ healthy controls (CDR = 0) drawn from the Alzheimer’s Disease Neuroimaging Initiative database (ADNI, <http://adni.loni.usc.edu/>), they found that the mild AD group had a mean brain-PAD of 10 years, whereas the control group mean brain-PAD was close to zero, supporting the notion that brain changes in AD may resemble accelerated brain aging.

Cole, Poudel, Tsagkrasoulis, Caan, Steves, Spector & Montana (2017) used deep learning, specifically convolutional neural networks, with raw and preprocessed T1-weighted MRI data to predict age in a sample of healthy adults aged 18 to 90 years old. They randomly split the data into training ($n = 1601$), validation ($n = 200$), and test ($n = 200$) sets. Using grey matter density maps, they report a correlation of $r = 0.96$ and MAE of 4.16 years for the test set. Performance was similar although slightly worse using Gaussian process regression ($r = 0.95$, MAE = 4.66 years).

Liem et al. (2017) showed that better age prediction can be attained when structural and functional imaging is combined. Structural data consisted of cortical thickness, cortical surface area, and subcortical volume measures, and functional data consisted of functional connectivity matrices using 197 and 444 regions. Linear support vector regression models were used to predict age using single modalities, and these models were then stacked in a random forest model to produce multimodal age predictions. The methods were applied to data from a randomly selected community-dwelling sample of $n = 2354$ volunteers aged 19 to 82 years, randomly split into a training set of $n = 1177$ and test set of $n = 1177$. The authors report a test set correlation of $r = 0.93$ and MAE of 4.29 years for the stacked multimodal model. Furthermore, they found that brain-PAD was associated with cognitive impairment.

Varikuti et al. (2018) used two study samples for model training, a population-based 1000BRAINS sample of $n = 693$ healthy older adults aged 55 to 75 years, and a MIXED dataset pooled from several sources of $n = 1084$ healthy adults aged 18 to 81 years. Using grey matter density maps, they first performed an orthonormal projective non-negative matrix factorization data reduction method. Then they applied lasso linear regression and relevance vector regression to predict age, using an iterated 10-fold cross-validation method to evaluate prediction performance. Several notable findings emerged from this study. First, voxel-wise lasso regression on the uncompressed data resulted in reasonably good predictions ($r = 0.69$ and $\text{MAE} = 3.4$ years for 1000BRAINS, $r = 0.91$ and $\text{MAE} = 4.9$ years for MIXED), but selected isolated voxels and thus produced brain maps more difficult to interpret than those derived from the reduced data. Secondly, lasso regression tended to outperform relevance vector regression when predicting age across samples, while performance was similar when predicting within samples. Thirdly, data trained on the older 1000BRAINS dataset did not extrapolate well to younger subjects in the MIXED sample. Finally, regions selected by models tended to be those most highly correlated with age. For 1000BRAINS, the authors report a correlation of $r = 0.65$ and MAE of 3.6 years, and for MIXED, $r = 0.88$ and MAE of 6.1 years.

3.1.3 Study objectives

Machine learning methods such as support vector regression and deep learning have yielded good accuracy for predicting age, but interpretation of the models can be challenging. Of the four studies reviewed above, only Franke et al. (2010) and Varikuti et al. (2018) provide brain maps of the estimated weights used in the models. However in the case of Franke et al. (2010), the model is not sparse. The models in Varikuti et al. (2018) are sparse, but require a data reduction step that may be computationally prohibitive if done separately for each cross-validation fold, or lead to an overly optimistic prediction error bias if performed on the entire training set. In the following sections we use fused sparse group lasso linear regression to predict age from voxel-wise multimodal imaging data. The method allows us to establish a parsimonious neuroimaging biomarker for age based on the estimated model coefficients.

We examine two imaging modalities: (1) grey matter density maps and (2) combined maps of the default mode and anterior salience network derived from resting state fMRI, to predict age both individually in unimodal models and together in a multimodal model. We compare various sparse and structured sparse penalties, including ridge, lasso, adaptive lasso, group lasso, and non-adaptive and adaptive versions of the fused sparse group lasso. We hypothesized that the structured penalties would offer improvement over unstructured penalties, and the multimodal model would perform better than either unimodal model.

We address the following questions:

- (1) How does each imaging modality perform separately for predicting age?
- (2) Does model performance improve when modalities are combined?
- (3) What is the best penalty structure in each case? Are adaptive weights helpful?
- (4) Which features are most predictive, i.e., what are the neuroimage changes associated with age?

3.2 MULTIMODAL FUSED SPARSE GROUP LASSO

3.2.1 Model

As described in section 2.2.1, assume we observe $\{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$ from n independent subjects, indexed by $i = 1, \dots, n$, where $y_i \in \mathbb{R}$ and $\mathbf{x}_i \in \mathbb{R}^p$. In a multimodal neuroimaging setting, each \mathbf{x}_i is a concatenated vector of intensity values from a set of q three dimensional brain images, indexed by $k = 1, \dots, q$, such that $\mathbf{x}_i = (\mathbf{x}_i^1, \mathbf{x}_i^2, \dots, \mathbf{x}_i^q)^T$. Each element of \mathbf{x}_i corresponds to one of p combined total voxels, where $p = \sum_{k=1}^q p_k$. The voxels do not necessarily have to be the same size within or across modalities. Assume that $\mathbf{y} = (y_1, \dots, y_n)$ and the columns of the matrix $\mathbf{X} = (\mathbf{x}_1 | \dots | \mathbf{x}_n)$ are centered, so we do not have an intercept term, and columns of \mathbf{X} are each rescaled to have a standard deviation of one. We model

the continuous outcome using standard linear regression,

$$\begin{aligned}\mathbf{y} &= \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon} \\ &= \sum_{k=1}^q \mathbf{X}_k \boldsymbol{\beta}_k + \boldsymbol{\epsilon}\end{aligned}\tag{3.1}$$

where $E(\boldsymbol{\epsilon}) = \mathbf{0}$, and $E(\boldsymbol{\epsilon}^T \boldsymbol{\epsilon}) = \sigma^2 \mathbf{I}_n$.

3.2.2 Estimator

There are a number of ways one could employ the fused sparse group lasso penalty in the multimodal imaging context, the choice of which will depend upon the research questions of interest and knowledge about the dependence structure of the predictors. Figure 3.1 shows a few possibilities. In all of the depicted scenarios, the fusion penalty is applied to neighboring voxels within the same modality, rather than across modalities. This is because, given that different imaging modalities measure different structural and functional characteristics, we do not generally expect the magnitude of coefficients to be similar across modalities in the same region. However, if the research question of interest concerns which brain regions are most predictive of the outcome, we could apply the group penalty term to voxels across modalities in the same region (Figure 3.1C). Alternatively, if the research question concerns which subset of image modalities is most predictive, then we might group together all voxels within the same modality (Figure 3.1D), or further subdivide voxels within modalities into smaller groups (Figure 3.1E). Finally, we could take a data-driven approach and group voxels based on cluster analysis, without regard to spatial location or modality (Figure 3.1F).

Regardless of the group structure we choose, the fused sparse group lasso estimator may be written in a manner similar to Equation 2.3,

$$\hat{\boldsymbol{\beta}} = \arg \min_{\boldsymbol{\beta} \in \mathbb{R}^p} L(\boldsymbol{\beta}) + \lambda_1 \|\boldsymbol{\beta}\|_1 + \lambda_2 \|\mathbf{D}\boldsymbol{\beta}\|_1 + \lambda_3 \Omega^{\mathcal{G}}(\boldsymbol{\beta}),\tag{3.2}$$

but now the finite difference matrix $\mathbf{D}_{m \times p}$ for the three dimensional fused lasso penalty term does not fuse across modalities, so

$$\|\mathbf{D}\boldsymbol{\beta}\|_1 = \sum_{k=1}^q \|\mathbf{D}_k \boldsymbol{\beta}_k\|_1.$$

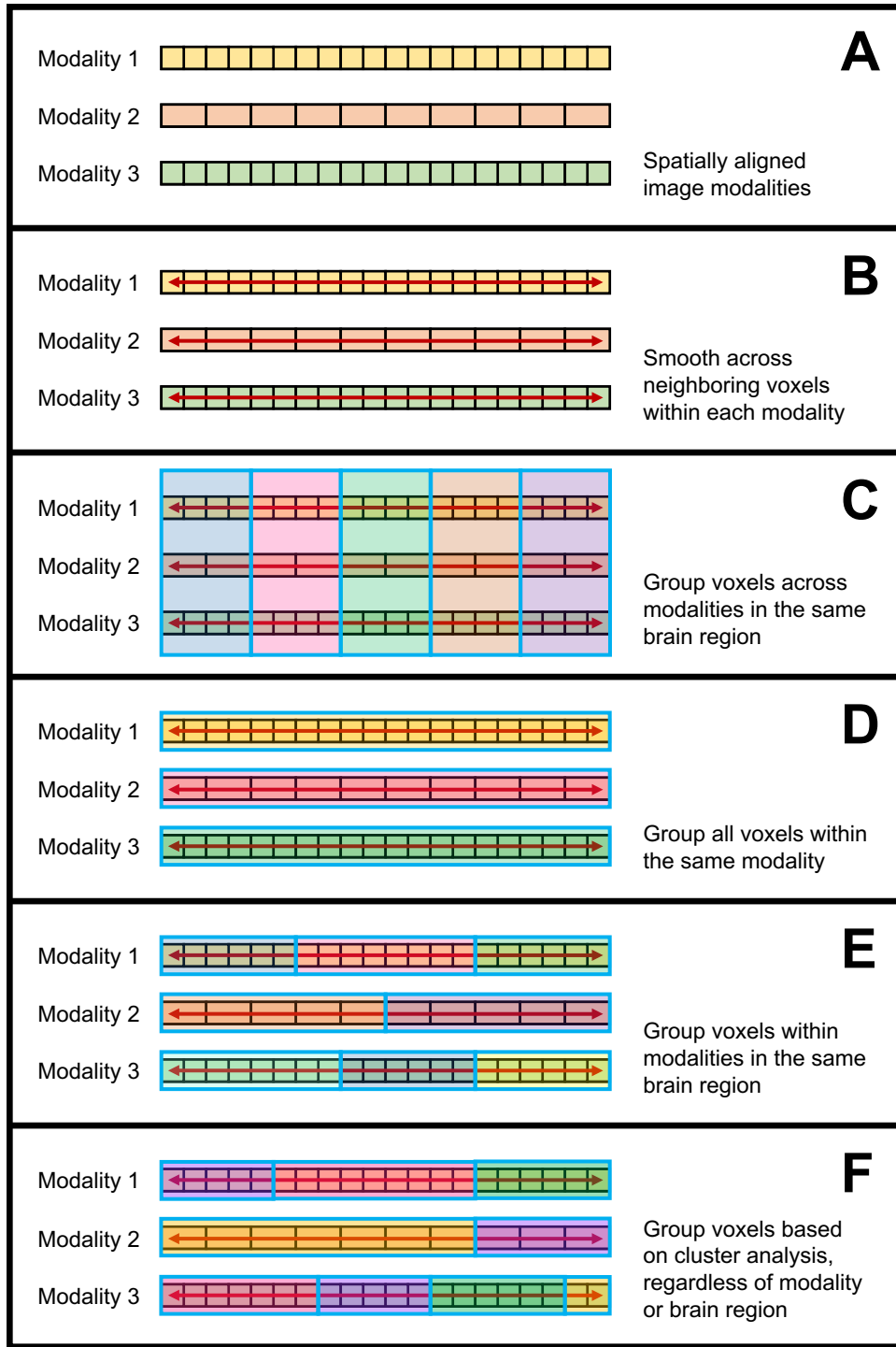


Figure 3.1: Possible variations of the fused sparse group lasso penalty for multimodal data

3.3 APPLICATION TO NORMAL AGING MULTIMODAL NEUROIMAGING DATA

3.3.1 Normal Aging dataset

A sample of $n = 71$ community dwelling, cognitively normal, volunteer participants were recruited via advertisements and mailings directed to individuals interested in aging research (mean (SD) age 76.9 (6.5) years, range 66 to 96 years, 48 (67.6%) female, see Table 3.1 for a descriptive summary and Figure 3.2 for age distribution). Participants were excluded if they had MCI or dementia, history of major neurological or psychiatric disease, Geriatric Depression Scale (Yesavage et al. 1982) score greater than 15, psychoactive medication use, or contraindications to MRI. A subset of this sample was included in the analyses reported in Karim et al. (2018) (currently under review).

Table 3.1: Normal Aging dataset descriptive summary

	Overall (n = 71)	Training set (n = 57)	Test set (n = 14)	<i>p</i> -value *
Age (mean (sd) [range])	76.9 (6.5) [66, 96]	76.9 (6.6) [66, 96]	76.9 (6.5) [67, 88]	0.908
Female (n (%))	48 (67.6)	39 (68.4)	9 (64.3)	0.759
Race (n (%))				1.000
White	60 (84.5)	48 (84.2)	12 (85.7)	
Black	10 (14.1)	8 (14.0)	2 (14.3)	
Asian	1 (1.4)	1 (1.8)	0 (0.0)	
Years education (mean (sd) [range])	14.8 (2.5) [12, 20]	14.9 (2.5) [12, 20]	14.5 (2.4) [12, 20]	0.516
Weight in pounds (mean (sd) [range])	160.5 (28.0) [103.4, 238.0]	161.2 (27.0) [105.8, 235.4]	157.6 (32.3) [103.4, 238.0]	0.520
PiB SUVR (mean (sd) [range])	1.7 (0.6) [1.2, 4.2]	1.7 (0.6) [1.2, 3.9]	1.6 (0.8) [1.2, 4.2]	0.902

* Wilcoxon rank test for continuous variables, Fisher's exact test for categorical variables

PiB: Pittsburgh Compound B, SUVR: standardized uptake value ratio

3.3.2 Methods

Participants underwent various structural MRI scans, task-based and resting state functional MRI scans, a PET scan using Pittsburgh Compound-B (PiB) to assess brain amyloid beta load, and also underwent neuropsychological assessment (see [Karim et al. \(2018\)](#) for further details). For the present study, we focused on grey matter density maps derived from T1-weighted MRI, and DMN/ASN maps derived from the resting state functional scans. For participants who underwent scans at multiple time points, we used the most recent time point at which both modalities were available.

3.3.2.1 MRI data collection A 3T Siemens Trio TIM scanner and 12-channel head coil were used to obtain T1-weighted MRI data via a sagittal whole brain MPRAGE (magnetization prepared rapid acquisition gradient echo) sequence (echo time (TE) = 2.98 ms, repetition time (TR) = 2300 ms, flip angle (FA) = 9°, field of view (FOV) = 256 × 240, 1 × 1 × 1.2 mm resolution, 0.6 mm gap, and GeneRalized Autocalibrating Partial Parallel Acquisition (GRAPPA) acceleration factor = 2). We collected an axial, whole brain (excluding cerebellum) resting state functional scan for 5 minutes (TE = 32 ms, TR = 2000 ms, FA = 90°, FOV = 128 × 128, 2 × 2 × 4 mm resolution, no gap, GRAPPA = 2, and 150 volumes). Due to low coverage and placement issues we had no coverage of the cerebellum or top of the motor/supplemental motor cortex.

3.3.2.2 MRI data preprocessing Data were preprocessed using Statistical Parametric Mapping (SPM12) software ([Ashburner et al. 2014](#)). Any image-based interpolation between images was done using fourth degree B-spline method. A rigid motion correction algorithm was applied using a mean reference and mutual information similarity metric. Structural MRI was then coregistered to the mean functional image using an affine transformation and normalized mutual information similarity metric. The structural image was bias corrected, segmented, and the resulting deformation field was output and used to normalize the functional images to Montreal Neurological Institute (MNI) space (2 mm isotropic resolution). The functional data were smoothed using a Gaussian kernel (FWHM 8 mm).

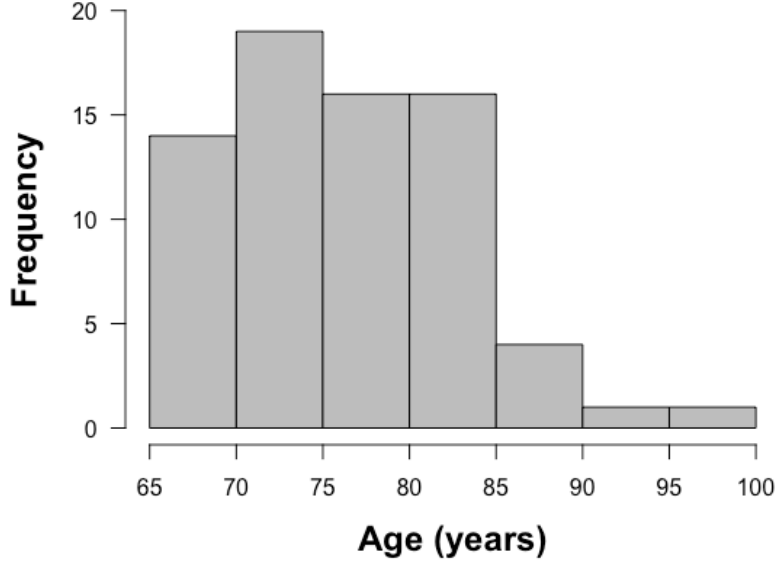


Figure 3.2: Age distribution for Normal Aging dataset

Grey matter density (GMD) maps were derived from the T1-weighted images using the SPM `segment` function, which implements an algorithm for probabilistic tissue classification involving alternating classification, bias correction, and image registration steps (Ashburner & Friston 2005). For the resting state data, maps of the DMN/ASN were created using template-based rotation (Schultz et al. 2014). GMD maps were resampled to the same voxel size in the resting state maps. We restricted our analyses to 235,706 voxels in the whole brain mask.

3.3.2.3 Feature screening See Figure 3.3 for an overview of the following methods. Participants were divided into training ($n = 57$) and test ($n = 14$) sets using an age-stratified randomization: Participants were ordered according to their age. Then random permutations of the numbers 1 to 5 were concatenated to split the data into five groups, and one group was selected to serve as the test set. We performed univariate statistical testing to assess whether baseline characteristics differed between training and test sets (Table 3.1).

For computational efficiency and to create sparse, interpretable models, we performed

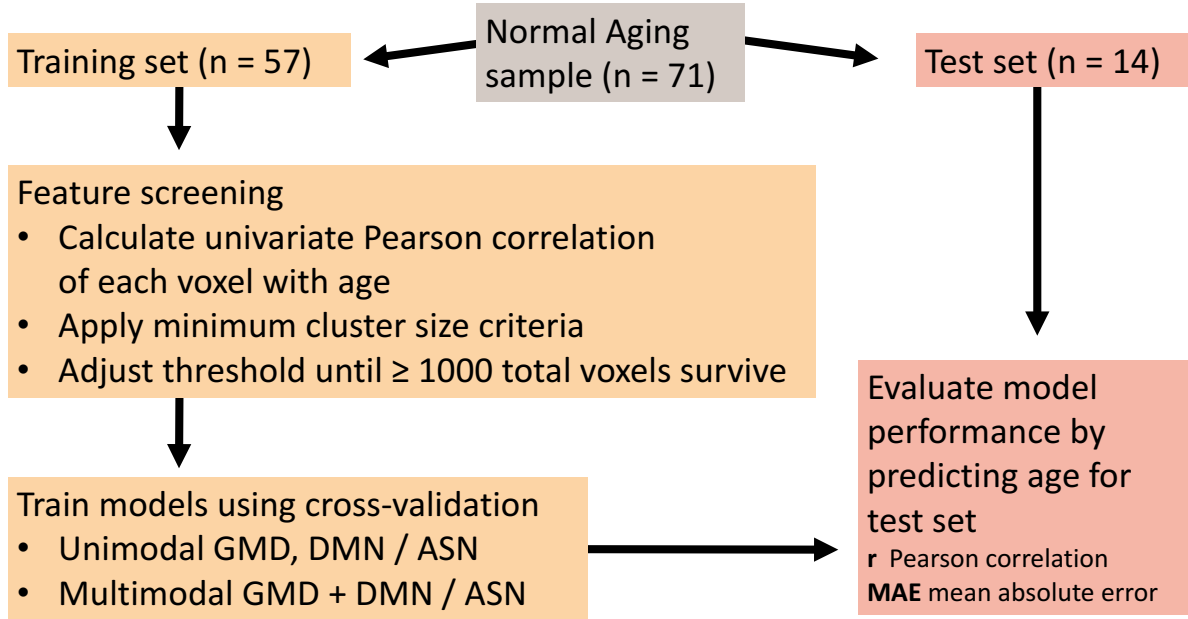


Figure 3.3: Overview of Normal Aging study data analysis methods

a feature screening step separately for each modality. In the training data, for each of the 235,706 voxels in the brain mask, we calculated univariate Pearson correlation with age. To enhance interpretability and help filter out noise voxels that might exhibit spurious correlations, we then imposed a minimum cluster size criteria. Exploratory analyses showed that a cluster size of 10 voxels for the GMD data and 50 voxels for the DMN/ASN data worked well. Finally, we adjusted the correlation thresholds for each modality until at least 1000 voxels survived. These steps were done using a combination of R version 3.4.0 ([R Core Team 2017](#)) (to calculate Pearson correlations), MATLAB version 9.1.0 (R2016b) ([MATLAB 2016](#)) with SPM12 software ([Ashburner et al. 2014](#)) (to transform raw Pearson correlation into NIFTI image files), and AFNI version 17.1.03 ([Cox 1996](#)) (to create spatial clusters).

3.3.2.4 Unimodal prediction models We created unimodal models predicting age for GMD and DMN/ASN separately. In the training set data, each column of the imaging data predictor matrices and the age outcome were standardized to have mean zero and standard

deviation of one. For each modality, we fit linear regression models with ridge, lasso, and adaptive lasso (using ridge regression coefficients to inversely weight the tuning parameter) penalties using R package `glmnet` (Friedman et al. 2010), and the group lasso penalty using R package `gglasso` (Yang & Zou 2015). Voxel groups were defined as spatially contiguous clusters, i.e., where each voxel is touching faces with at least one neighboring voxel. We then fit eight different fused sparse group lasso (FSGL) models, including one set of nonadaptive penalties, with (α, γ) set equal to $(0, 1)$ (group lasso), $(0.2, 1)$ (sparse group lasso), $(0, 0.8)$ (fused group lasso), and $(0.1, 0.9)$ (fused sparse group lasso), and one set of adaptive FSGL models using the same set of (α, γ) combinations and weights defined using estimated ridge regression coefficients as defined in 2.2.4. Five-fold cross-validation was used to select the value for λ that achieved the minimum cross-validation error. Cross-validation folds were age-stratified using the same method that was used to define training and test sets, and folds were the same across all model fitting. Once the optimal λ value was selected, final models were fit on the full training set. To evaluate model sparsity, we noted how many coefficients were nonzero for the `glmnet` and `gglasso` models. The ADMM algorithm used to fit the FSGL models does not produce estimated coefficients that are identically zero, so we chose the threshold ≥ 0.0001 to evaluate relative sparsity for these. For the GMD data, the ADMM tolerance levels were set to $\epsilon_{\text{abs}} = \epsilon_{\text{rel}} = 10^{-8}$, step-size was set to $\rho = 1$, and the maximum number of iterations was set to 8000. For the DMN/ASN data, the ADMM tolerance levels were set to $\epsilon_{\text{abs}} = \epsilon_{\text{rel}} = 10^{-8}$, step-size was set to $\rho = 10$, and the maximum number of iterations was set to 6000. The `glmnet` and `gglasso` models were fit using R version 3.4.0 (R Core Team 2017) and the FSGL models were fit using MATLAB version 9.1.0 (R2016b) (MATLAB 2016) and SPM12 software (Ashburner et al. 2014).

The estimated models were evaluated on the test set data. Columns of the test set predictor matrices and the age outcome were standardized using the training set means and standard deviations, and predicted age for each individual was obtained using the estimated coefficients. We transformed the age estimates back to the original scale and used Pearson correlation (r), root mean squared error ($\text{RMSE} = \|\hat{\mathbf{y}}_{\text{test}} - \mathbf{y}_{\text{test}}\|_2/14$), and mean absolute error ($\text{MAE} = \|\hat{\mathbf{y}}_{\text{test}} - \mathbf{y}_{\text{test}}\|_1/14$) to evaluate prediction performance. To assess whether models showed systematic age-related bias, we examined the correlation of residuals (i.e.,

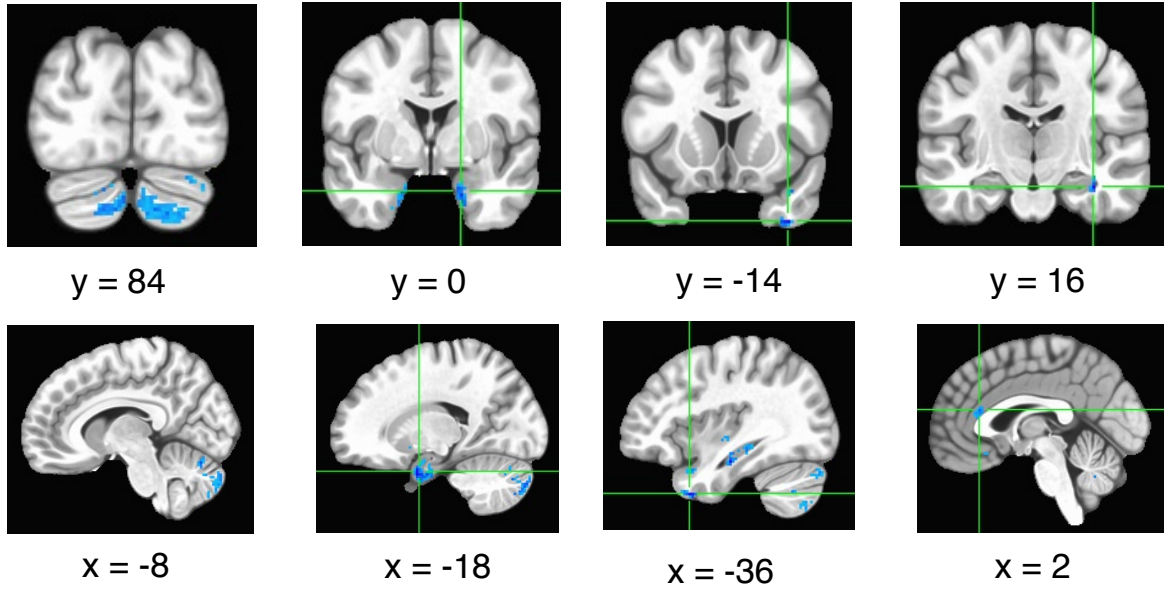
brain-PAD) from the best models with actual age. To assess whether predicted age was consistently younger or older than chronological age in both modalities for a given individual, we examined the correlation between the residuals (age-adjusted to remove any systematic bias) from the best models in each modality.

3.3.2.5 Multimodal prediction models A similar procedure was applied to training and test set data for the multimodal models as was done for the unimodal models, only in this case the GMD and DMN/ASN data was concatenated. For the FSGL penalties, fusion was applied only within modalities. Likewise, groups were defined as spatially contiguous clusters within the same modality, i.e., the group structure depicted in Figure 3.1E. ADMM tolerance levels were set to $\epsilon_{\text{abs}} = \epsilon_{\text{rel}} = 10^{-8}$, step-size was set to $\rho = 10$, and the maximum number of iterations was set to 8000. In addition to concatenation of the GMD and DMN/ASN data, we also examined the performance of age predictions obtained by averaging the predicted age from the best unimodal models for each modality.

3.3.3 Results

3.3.3.1 Feature screening Feature screening results are shown in Figure 3.4 and Appendix Tables C1 and C2. For the GMD data, a Pearson correlation threshold of $|r| \geq 0.4145$ and minimum cluster size of 10 voxels yielded 30 clusters of 1009 total voxels. All GMD clusters showed a negative correlation with age. Fifteen clusters occurred in the cerebellum, including the largest cluster. Other regions included the right and left parahippocampal gyrus, right medial temporal lobe, and right hippocampus. Full results for GMD data are listed in Appendix Table C1. For the DMN/ASN data, a threshold of $|r| \geq 0.3363$ and minimum cluster size of 50 voxels yielded 9 clusters of 1000 total voxels. The largest clusters were at the left anterior cingulate cortex, left medial frontal gyrus, and left cingulate gyrus. All DMN/ASN clusters showed a negative correlation with age except for the right middle occipital/inferior temporal gyrus and right superior parietal lobule. Full results for DMN/ASN data are listed in Appendix Table C2.

A Grey matter density



B Default mode / anterior salience network

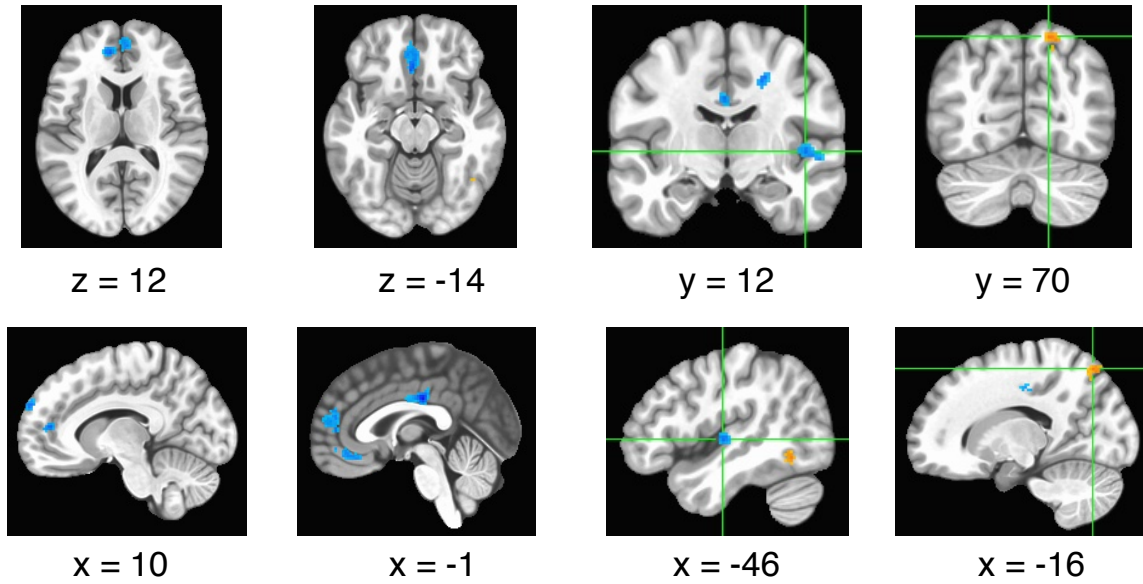


Figure 3.4: Feature screening results for GMD and DMN/ASN data. Univariate feature screening results for predicting age using (A) grey matter density (GMD) and (B) default mode/anterior salience network (DMN/ASN) data. For the GMD data, a Pearson correlation threshold of $|r| \geq 0.4145$ and minimum cluster size of 10 voxels yielded 30 clusters of 1009 total voxels. For the DMN/ASN data, a threshold of $|r| \geq 0.3363$ and minimum cluster size of 50 voxels yielded 9 clusters of 1000 total voxels. Blue represents negative and orange represents positive correlation. MNI coordinates are shown below each image.

3.3.3.2 Unimodal prediction models Cross-validation curves for all GMD and DMN/ASN unimodal models are presented in Appendix Figures C1 and C2, plots of estimated coefficients versus ridge regression coefficients are presented in Appendix Figures C4 and C5, and plots of predicted versus actual age for the test set data are presented in Appendix Figures C7 and C8.

For the GMD data, the smallest test set RMSE (5.01) and MAE (3.91 years) was achieved by the adaptive fused group lasso penalty ($\alpha = 0, \gamma = 0.8$), while the highest Pearson ($r = 0.646$) and Spearman ($\rho = 0.633$) correlations were achieved by the lasso penalty (Table 3.2). The lasso model was rather sparse, however, with only 50 nonzero coefficients, which limits its interpretability. Therefore, we selected the adaptive fused group lasso as the best model for GMD, as it had 902 coefficients ≥ 0.0001 , and the third highest Pearson ($r = 0.621$) and Spearman ($\rho = 0.529$) correlations. The brain map of estimated coefficients is shown in Figure 3.5, and a plot of predicted versus actual age for the test set is shown in Figure 3.7A.

For the DMN/ASN data, the smallest test set RMSE (4.09) and MAE (3.48 years) and the highest Pearson ($r = 0.780$) and Spearman ($\rho = 0.752$) correlations were achieved by the adaptive group lasso penalty ($\alpha = 0, \gamma = 1$) (Table 3.3). This model had 733 coefficients ≥ 0.0001 . The brain map of estimated coefficients is shown in Figure 3.6, and a plot of predicted versus actual age for the test set is shown in Figure 3.7B.

Table 3.2: Comparison of estimators applied to Normal Aging dataset: GMD data

Method	α	γ	Estimator	Training set (n = 57)		Test set (n = 14)				
				Optimal λ	CV MSE (SE)	RMSE	MAE	Pearson $r(p)$	Spearman $\rho(p)$	Nonzero coef**
Glmnet			Ridge	9.569	0.298 (0.051)	6.02	4.98	0.406 (0.150)	0.326 (0.255)	1009
			Lasso	0.012	0.263 (0.041)	5.40	4.33	0.646 (0.013)	0.633 (0.015)	50
			Adaptive lasso	0.585	0.134 (0.034)	5.34	3.98	0.641 (0.013)	0.582 (0.029)	34
Gglasso			Group lasso	0.061	0.316 (0.058)	5.54	4.48	0.525 (0.054)	0.459 (0.099)	457
FSGL	0.0	1.0	Group lasso	0.594	0.301 (0.043)	5.25	4.25	0.586 (0.028)	0.474 (0.087)	660
	0.2	1.0	Sparse group lasso	0.594	0.289 (0.040)	5.27	4.24	0.582 (0.029)	0.474 (0.087)	625
	0.0	0.8	Fused group lasso	1.190	0.295 (0.050)	5.34	4.29	0.562 (0.037)	0.481 (0.082)	616
	0.1	0.9	Fused sparse group lasso	1.190	0.294 (0.050)	5.38	4.33	0.553 (0.041)	0.481 (0.082)	550
Adaptive FSGL	0.0	1.0	Adaptive group lasso	9.591	0.303 (0.038)	5.47	4.40	0.553 (0.040)	0.434 (0.121)	944
	0.2	1.0	Adaptive sparse group lasso	13.581	0.138 (0.018)	5.52	4.47	0.544 (0.044)	0.454 (0.103)	261
	0.0	0.8	Adaptive fused group lasso	13.581	0.196 (0.024)	5.01	3.91	0.621 (0.018)	0.529 (0.052)	902
	0.1	0.9	Adaptive fused sparse group lasso	19.231	0.105 (0.011)	5.37	4.43	0.568 (0.034)	0.450 (0.107)	260

** Nonzero coef: number of nonzero estimated coefficients for Glmnet and Gglasso, or ≥ 0.0001 for FSGL and Adaptive FSGL

Table 3.3: Comparison of estimators applied to Normal Aging dataset: DMN/ASN data

Method	α	γ	Estimator	Training set (n = 57)		Test set (n = 14)				
				Optimal λ	CV MSE (SE)	RMSE	MAE	Pearson $r(p)$	Spearman $\rho(p)$	Nonzero coef**
Glmnet			Ridge	61.979	0.488 (0.065)	4.80	4.27	0.690 (0.006)	0.701 (0.005)	1000
			Lasso	0.228	0.824 (0.084)	5.13	4.37	0.716 (0.004)	0.613 (0.020)	17
			Adaptive lasso	0.312	0.635 (0.066)	4.64	3.92	0.754 (0.002)	0.745 (0.002)	25
Gglasso			Group lasso	0.081	0.605 (0.117)	4.69	4.18	0.717 (0.004)	0.741 (0.002)	1000
FSGL	0.0	1.0	Group lasso	3.378	0.600 (0.057)	4.61	4.13	0.713 (0.004)	0.701 (0.005)	979
	0.2	1.0	Sparse group lasso	3.378	0.611 (0.056)	4.60	4.12	0.713 (0.004)	0.659 (0.007)	895
	0.0	0.8	Fused group lasso	4.784	0.577 (0.054)	4.69	4.20	0.710 (0.005)	0.692 (0.006)	999
	0.1	0.9	Fused sparse group lasso	4.784	0.595 (0.055)	4.70	4.20	0.712 (0.004)	0.701 (0.005)	992
Adaptive FSGL	0.0	1.0	Adaptive group lasso	867.797	0.598 (0.088)	4.09	3.48	0.780 (0.001)	0.752 (0.002)	753
	0.2	1.0	Adaptive sparse group lasso	52.324	0.629 (0.069)	4.72	4.07	0.675 (0.008)	0.723 (0.004)	82
	0.0	0.8	Adaptive fused group lasso	340.286	0.519 (0.077)	4.41	3.92	0.737 (0.003)	0.703 (0.005)	996
	0.1	0.9	Adaptive fused sparse group lasso	52.324	0.522 (0.061)	4.97	4.45	0.640 (0.014)	0.602 (0.023)	929

** Nonzero coef: number of nonzero estimated coefficients for Glmnet and Gglasso, or ≥ 0.0001 for FSGL and Adaptive FSGL

Grey matter density

Estimated coefficients from best unimodal model

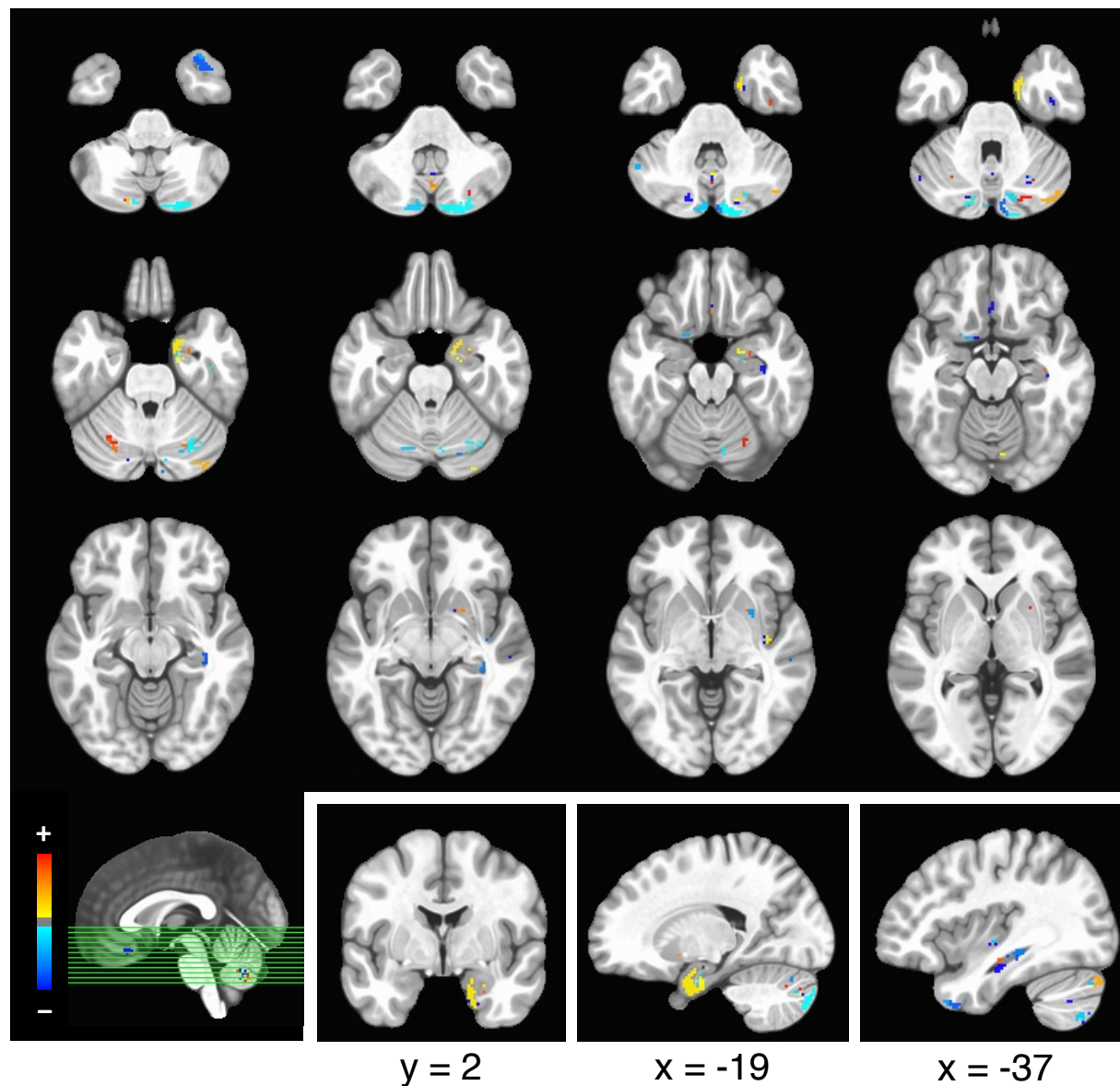


Figure 3.5: Estimated coefficients for best GMD unimodal model. Axial brain slices in the top three rows are from the MNI z -coordinates shown at lower left. MNI coordinates are given below images at bottom right.

Default mode / anterior salience network
 Estimated coefficients from best unimodal model

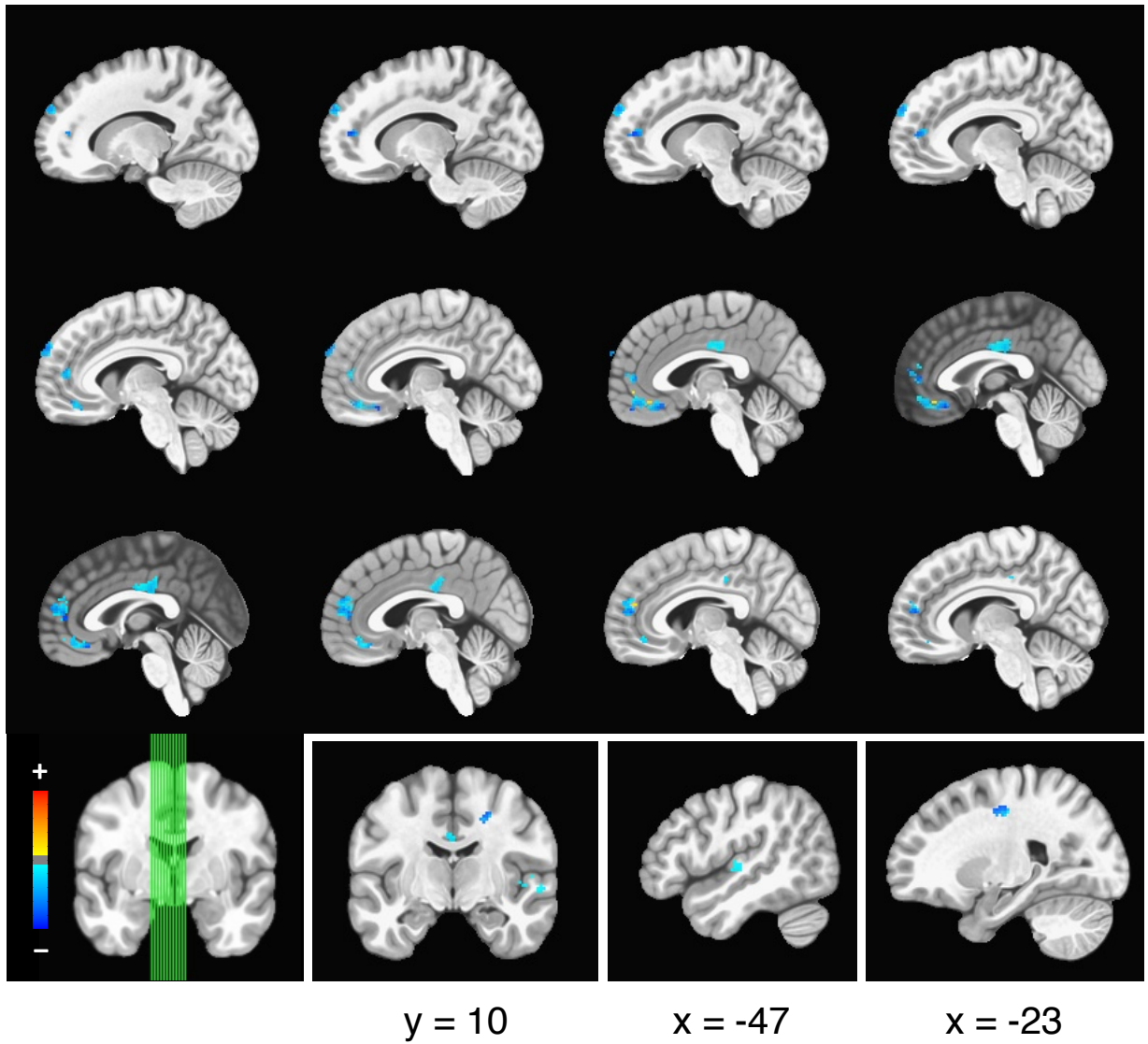


Figure 3.6: Estimated coefficients for best DMN/ASN unimodal model. Sagittal brain slices in the top three rows are from the MNI x -coordinates shown at lower left. MNI coordinates are given below images at bottom right.

3.3.3.3 Multimodal prediction models Cross-validation curves for all multimodal models are presented in Appendix Figure C3, plots of estimated coefficients versus ridge regression coefficients are presented in Appendix Figure C6, and plots of predicted versus actual age for the test set data are presented in Appendix Figure C9.

The best multimodal model results were given by the lasso penalty, which had RMSE = 4.85, MAE = 3.76 years, $r = 0.690$, and $\rho = 0.617$ (Table 3.4, Figure 3.7C). The multimodal lasso performed slightly better than the best GMD model, but worse than the best DMN/ASN model. This model had only 54 nonzero coefficients. Adaptive lasso yielded the second-best performance, but was also sparse, with 43 nonzero coefficients. The best structured penalty performance was given by the adaptive fused group lasso ($\alpha = 0, \gamma = 0.8$), which had RMSE = 5.01, MAE = 3.80 years, $r = 0.622$, $\rho = 0.481$, and 983 coefficients ≥ 0.0001 . Predictions for this model were almost identical to those for the GMD unimodal adaptive fused group lasso ($\alpha = 0, \gamma = 0.8$) ($r = 0.996$, Appendix Figures C7 and C9).

Averaging the predictions from the best unimodal models yielded better results, with RMSE = 4.11, MAE = 3.22 years, $r = 0.778$, and $\rho = 0.763$, which is comparable to the best DMN model (Figure 3.7D).

Table 3.4: Comparison of estimators applied to Normal Aging dataset: Multimodal GMD and DMN/ASN data

Method	α	γ	Estimator	Training set (n = 57)		Test set (n = 14)				
				Optimal λ	CV MSE (SE)	RMSE	MAE	Pearson $r(p)$	Spearman $\rho(p)$	Nonzero coef**
Glmnet			Ridge	67.510	0.267 (0.040)	5.27	4.62	0.550 (0.042)	0.501 (0.068)	2009
			Lasso	0.013	0.435 (0.051)	4.85	3.76	0.690 (0.006)	0.617 (0.019)	54
			Adaptive lasso	0.067	0.158 (0.018)	5.02	3.77	0.674 (0.008)	0.617 (0.019)	43
Gglasso			Group lasso	0.041	0.371 (0.068)	5.13	4.13	0.592 (0.026)	0.500 (0.068)	1276
FSGL	0.0	1.0	Group lasso	1.237	0.339 (0.062)	5.02	4.02	0.614 (0.019)	0.501 (0.068)	1243
	0.2	1.0	Sparse group lasso	0.979	0.340 (0.063)	5.03	4.03	0.614 (0.020)	0.494 (0.073)	1154
	0.0	0.8	Fused group lasso	2.497	0.348 (0.067)	5.04	4.08	0.607 (0.022)	0.494 (0.073)	1283
	0.1	0.9	Fused sparse group lasso	1.563	0.343 (0.064)	4.99	4.01	0.618 (0.019)	0.501 (0.068)	1177
Adaptive FSGL	0.0	1.0	Adaptive group lasso	12.848	0.314 (0.039)	5.78	4.64	0.509 (0.063)	0.428 (0.127)	922
	0.2	1.0	Adaptive sparse group lasso	66.121	0.188 (0.022)	5.66	4.53	0.525 (0.054)	0.445 (0.111)	138
	0.0	0.8	Adaptive fused group lasso	8.045	0.248 (0.034)	5.01	3.80	0.622 (0.018)	0.481 (0.082)	983
	0.1	0.9	Adaptive fused sparse group lasso	32.765	0.191 (0.016)	6.14	5.10	0.504 (0.066)	0.452 (0.105)	228
Average of best GMD and DMN/ASN unimodal model predictions						4.11	3.22	0.778 (0.001)	0.763 (0.002)	

** Nonzero coef: number of nonzero estimated coefficients for Glmnet and Gglasso, or ≥ 0.0001 for FSGL and Adaptive FSGL

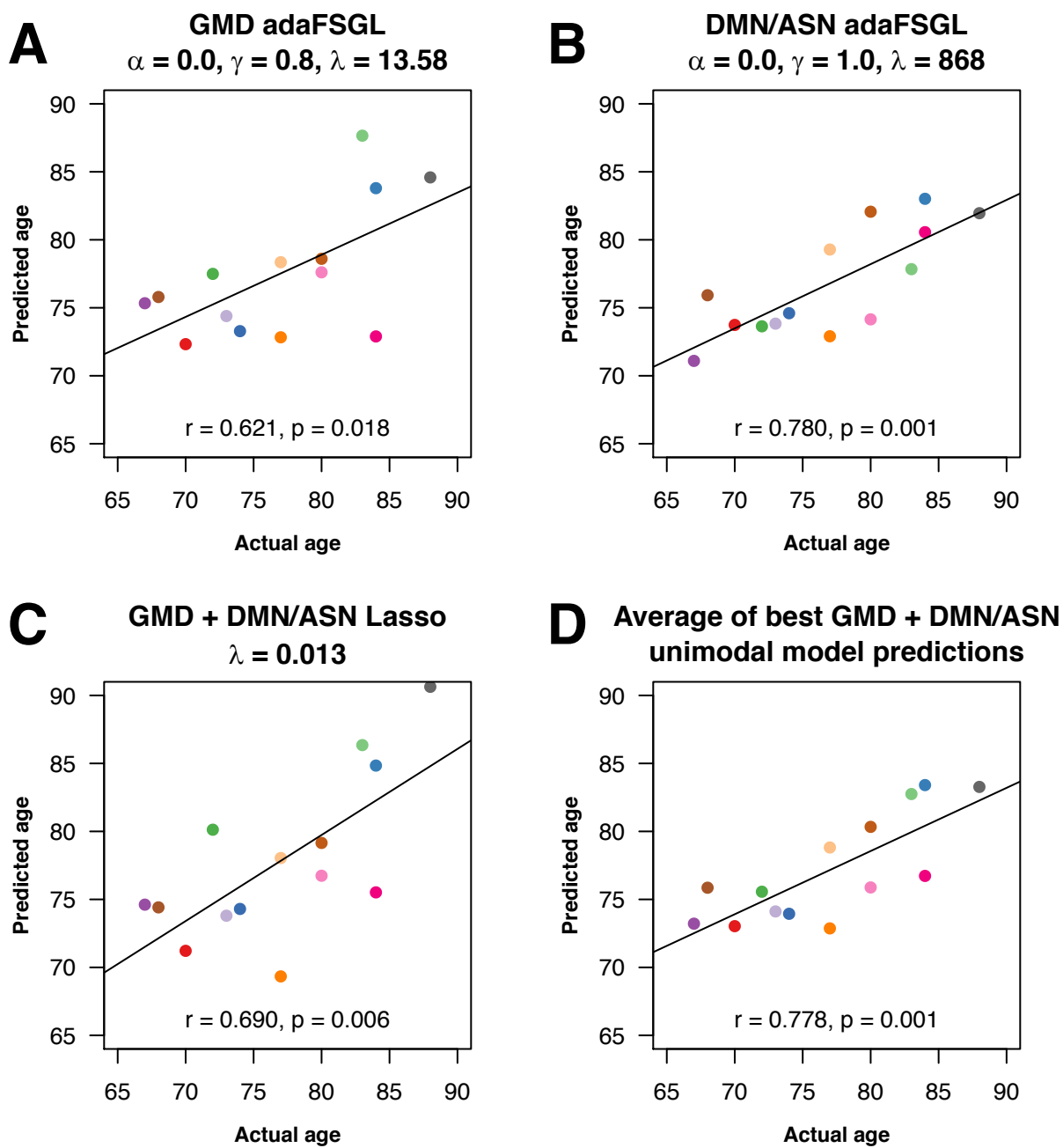


Figure 3.7: Predicted versus actual age for best unimodal and multimodal models

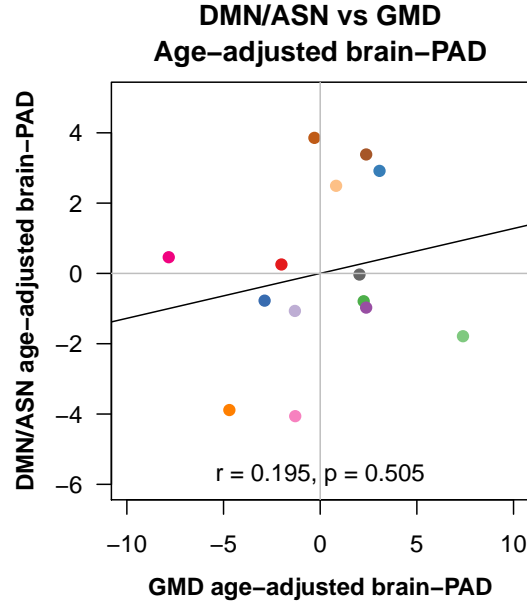


Figure 3.8: Correlation of age-adjusted brain-PAD for best unimodal models

3.3.3.4 Analysis of brain-PAD from best models All of the best unimodal and multimodal models showed a rather strong linear association of brain-PAD with actual age (Figure 3.9). Younger participants were more likely to have a positive brain-PAD, and older participants were more likely to have a negative brain-PAD. This is not an entirely unexpected result for penalized regression methods, as they can produce predictions biased towards zero as a result of the bias (also towards zero) imposed on the coefficient estimates. This suggests that for penalized linear regression methods, some adjustment of the brain-PAD for systematic age bias may be necessary to create a useful biomarker.

There was a weak positive correlation between the age-adjusted brain-PADs from the best models in each modality (Figure 3.8). Four of the 14 test set participants had a consistently negative adjusted brain-PAD, and 3 had consistently positive adjusted brain-PAD. Three participants had positive GMD adjusted brain-PAD and negative DMN/ASN adjusted brain-PAD, the remaining 4 participants had adjusted brain-PAD close to zero in one modality and nonzero in the other modality.

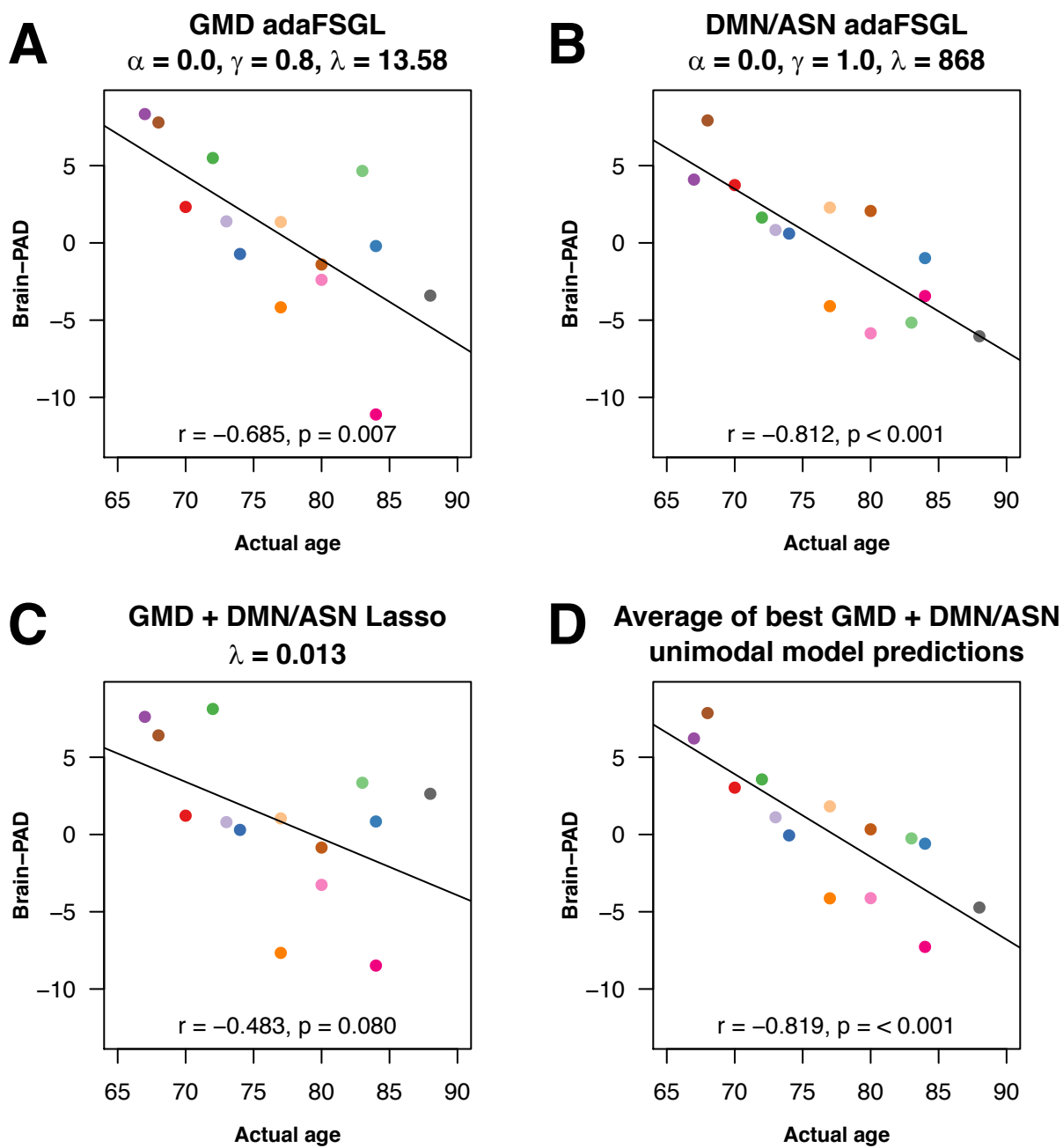


Figure 3.9: Brain-PAD versus actual age for best unimodal and multimodal models

3.4 DISCUSSION

In this chapter, we used unimodal and multimodal penalized linear regression models to predict chronological age using GMD and DMN/ASN neuroimaging data from a sample of $n = 71$ cognitively normal adults aged 66 to 96 years. We found that, somewhat unexpectedly, the functional DMN/ASN data produced better age predictions than the structural GMD data. Furthermore, the multimodal model with concatenated data did not do better than the best unimodal model, as we expected. Averaging the unimodal model predictions produced better results, yielding slightly lower MAE and higher Spearman correlation than the best DMN/ASN model, and also performing comparably with respect to RMSE and Pearson correlation.

Our results compare reasonably well with previously published studies, particularly regarding the low MAE (3.22 years for the averaged multimodal model), but the correlation ($r = 0.778$) was not as high as that reported in the best models from [Franke et al. \(2010\)](#), [Cole, Poudel, Tsagkrasoulis, Caan, Steves, Spector & Montana \(2017\)](#), [Liem et al. \(2017\)](#), [Varikuti et al. \(2018\)](#), which yielded correlations of $r > 0.9$. However, while those studies tended to use large lifespan samples ranging in age from 18 to 90 years, with training sets of hundreds or over 1000 participants, our sample size was small, with a training set of only $n = 57$ and test set of $n = 14$. Therefore our estimates are likely to be less precise. Furthermore, our sample had a more restricted age range, which contributes directly to reduced correlation ([Bland & Altman 2011](#)). An important finding from [Franke et al. \(2010\)](#) was that the size of the training set had a strong influence on prediction accuracy. When they used only 1/4 of the training set ($n = 103$) to fit the model, the MAE increased from 5 to 5.6 years. Other studies have also highlighted the large influence of sample size on prediction accuracy ([Cui & Gong 2018](#), [Chu et al. 2012](#)). Thus, we expect that the accuracy level would be likely to increase when using the proposed methods in a larger sample drawn from the same population.

The test set of $n = 14$ is particularly small and thus it is hard to determine whether conclusions will generalize. To make more efficient use of the dataset, nested cross-validation (as was used in [Varikuti et al. \(2018\)](#)) is one option, as it would produce a predicted age for

each individual in the sample. To avoid optimistic prediction error bias, the entire model fitting process, including the feature screening step, would need to be repeated in each fold of the data. This would make it essential to have a computationally efficient feature screening method. Although this would result in multiple estimated coefficient maps, the importance of regions for prediction could be assessed based on their frequency of selection. Another way to further validate the models would be to use them to predict age in another, preferably large, sample, such as from the ADNI dataset.

Since we screened out a large proportion of voxels (using only 1000/235,706 or 0.42% of voxels in the brain mask) before fitting the penalized models, feature screening is a critical step in the proposed method. Although the penalized regression did shrink a large proportion of the screened voxels to zero in some cases, in other cases it did not, so one might argue that the biomarker for age is driven largely by the feature screening step. This may not be too detrimental, as [Varikuti et al. \(2018\)](#), for example, noted that regions selected in their penalized models also tended to be those most correlated with age. However, univariate feature screening does not take into account the multivariate patterns in the data, and particularly since neuroimaging data tends to be highly correlated, selected features may provide redundant information. In our application, GMD data did not perform as well as DMN/ASN data, and this may be because the feature screening step did not select an optimal set of features for GMD. The fact that half of the screened clusters occurred in the cerebellum, and very sparse models such as lasso and adaptive lasso did better for the GMD data, lends support to the idea that much of the information in the selected features may be redundant. Furthermore, our choice of minimum cluster size and number of voxels to include was somewhat arbitrary. The feature screening step may be improved by doing a more systematic analysis of these choices, incorporating of regions of interest based on previous studies, or perhaps implementing a multivariate feature screening approach such as recursive feature elimination ([Mwangi et al. 2014](#)).

The brain regions predictive of age selected by our models are in agreement with some of the literature. For the DMN/ASN data, we found that decreased connectivity with anterior midline regions, particularly in the left hemisphere, were predictive of older age. This is consistent with findings from several studies which noted decreased resting state DMN activity

in older participants ([Damoiseaux et al. 2007](#), [Tomasi & Volkow 2012](#)), as well as findings in [Andrews-Hanna et al. \(2007\)](#) which found disruption of anterior to posterior connectivity of the DMN in older participants during a semantic classification task. For the GMD data, the abundance of cerebellar voxels selected was somewhat surprising. However, cerebellar atrophy has been consistently related to age in several studies, although the effect size is typically not large ([Fjell & Walhovd 2010](#)). We also found several regions that were more expected, such as left and right parahippocampal gyri, right hippocampus, several right temporal regions, and right putamen. We only found about 4 small clusters in frontal cortical regions (left and right inferior frontal gyri, right medial frontal gyrus, and left anterior cingulate cortex). Other frontal regions may have been screened out due to the large number of cerebellar regions that were included.

For the multimodal model, we found that concatenating modalities was not as effective as taking a simple average of the predictions from the two unimodal models. This may be because the method we used allows for only one set of tuning parameters, while the optimal degree of smoothness, sparsity structure, and optimal regularization level might vary across modalities. There may be better penalty weighting schemes that take this into account. Other studies have suggested that multi-kernel or ensemble methods may be a better way to combine heterogeneous data modalities. Both [Zhang et al. \(2011\)](#) and [Hinrichs et al. \(2011\)](#), which examined multimodal classification of MCI and AD, examined concatenation in comparison with their multi-kernel support vector machine methods, and found that the multi-kernel methods gave greater classification accuracy than concatenation.

Since accelerated brain aging has been found in a range of neuropsychiatric and other health conditions, the lack of specificity of predicted brain age poses a problem for differential diagnosis ([Cole et al. 2018](#)). Furthermore, the particular spatial pattern of changes seen in normal aging may not capture all changes seen in neuropsychiatric disorders and diseases. For example, in AD, neurodegeneration is prominent in the medial temporoparietal memory network, including hippocampus, entorhinal retrosplenial, posterior cingulate, and precuneus cortices, while normal aging, on the other hand, is typically marked by deterioration in frontostriatal regions, which is associated with executive dysfunction ([Fjell & Walhovd 2010](#)).

Therefore, methodological approaches accounting for heterogeneity of brain aging are likely to be a promising direction for future research (e.g., [Eavani et al. \(2018\)](#)).

As this study demonstrates, the structured, sparse fused sparse group lasso penalty can enhance prediction accuracy and interpretability when used in unimodal and multimodal models to predict age. For the GMD data, the adaptive fused group lasso gave the lowest RMSE and MAE, and for the DMN/ASN data, the adaptive group lasso performed best on all metrics. For the concatenated multimodal model, however, the lasso performed the best on all measures, but this may not be true if feature selection is performed in a more optimal way, particularly for the GMD data. Taking the average of the two best structured sparse unimodal models performed better. As noted in Section 3.2.2, there are many ways the fused sparse group lasso penalties could be implemented for multimodal neuroimaging data, and we only explored one possibility. Nevertheless, the performance we achieved is in line with previous age prediction results, particularly with regard to the MAE, with the added benefit of producing sparse, interpretable age biomarkers for GMD and DMN/ASN modalities.

4.0 CONCLUSIONS

Since the sparsity-inducing lasso penalty was introduced in 1996 (Tibshirani 1996), numerous penalty variations have been proposed, including the elastic net (Zou & Hastie 2005), fused lasso (Tibshirani et al. 2005), group lasso (Yuan & Lin 2006), and overlapping group lasso (Obozinski et al. 2011). One reason for the popularity of penalized methods is the recent explosion of high dimensional data. These methods allow model fitting in high dimensional data (specifically, when there are more predictors than observations), yielding a unique solution to an originally ill-posed problem by imposing constraints. The constraints can encode *a priori* expectations, e.g., that the model is sparse or the coefficients ought to have a particular structure such as grouping, spatial smoothness, or hierarchy (Bach et al. 2012, Zhao et al. 2009). Sparsity-inducing penalized methods can perform variable selection automatically and simultaneously as the model is fit. Furthermore, through the use of cross-validation to determine the optimal degree of regularization, penalized methods can prevent overfitting. If it is accurate, adding information about structure can potentially improve interpretability, coefficient estimation, and prediction accuracy.

In this dissertation we studied properties of the fused sparse group lasso, a generalization of the lasso, group lasso, and fused lasso. This penalty has been introduced previously (Zhou et al. 2012); however, in this work we specifically explore its application to the prediction of clinical variables from voxel-wise neuroimaging data. This is a unique setting because not only is the data high dimensional, but also, the predictors are all of the same type and have a three dimensional spatial structure. Additionally, there is almost always high levels of correlation between neuroimage predictors, which can pose a problem for standard lasso regularization. The novel contributions of this dissertation include an algorithm and flexible R and MATLAB functions to fit the fused sparse group lasso. In contrast to the software

provided in [Zhou et al. \(2011\)](#), our algorithm allows smoothing across groups, not only within groups. We introduced a convenient reparameterization of the tuning parameters for ease of cross-validation. Furthermore, we introduced the adaptive fused sparse group lasso, and found that it tended to perform best in applications to real neuroimaging data. To our knowledge, the fused sparse group lasso has never been systematically investigated using simulation studies. The simulation study presented in [Section 2.3](#) considered a range of spatial and group structures for the true model coefficients. We found that the combination of fusion and group penalties worked best in most of the scenarios we explored. Regarding the applications to neuroimaging data, we found that prediction of Social Responsiveness Scale scores from voxel-wise subcortical-cortical connectivity maps performed on a par with methods that use whole brain region of interest-based functional connectivity matrices. We also predicted age from grey matter density and default mode/anterior salience resting state network maps, and found that prediction from the functional maps did surprisingly well. Both of these applications produced brain maps of estimated coefficients that illustrate the most predictive brain regions.

Recently, many researchers have argued that neuroscience and psychology should move towards a focus on prediction over hypothesis testing or other more traditional types of statistical inference ([Woo et al. 2017](#), [Yarkoni & Westfall 2017](#)). Reasons for this include (1) predictive models allow for individual predictions, which could be useful for making diagnosis or treatment decisions in a clinical setting; (2) multivariable approaches can potentially explain more variation in the outcome of interest than a mass univariate approach, since they take into account the multivariate relationships in the predictors; (3) predictive approaches generally do not have the multiple testing problems that a mass univariate approach does, and they are less prone to p -hacking than traditional hypothesis testing; (4) predictive models can be validated on independent test datasets; (5) predictive methods may be more fruitful than explanatory methods because the underlying data generating process is too complex to model directly. However, the accuracy of predictive models can sometimes rely on confounding variables, such as head motion ([Eloyan et al. 2012](#)) — which does not require expensive neuroimaging to measure — rather than more intrinsic features of brain structure and function underlying the neuropsychiatric condition of interest. Therefore, it is

desirable to make the prediction mechanism as transparent as possible. Indeed, there is an emerging area known as *interpretable machine learning* dedicated to this goal. Many of the methods currently used to predict clinical variables from neuroimages are less interpretable than those explored in this dissertation. For example, deep learning has become extremely popular, shown to be effective for many problems involving prediction and classification, and increasingly applied to problems in neuroimaging, such as predicting age from neuroimages in [Cole, Poudel, Tsagkraloulis, Caan, Steves, Spector & Montana \(2017\)](#). Yet, not only is the resulting model difficult to interpret, such methods requires a large amount of data to tune their many parameters, and prediction performance may be only marginally improved over simpler more interpretable methods. This illustrates a more general principle described in [Hand \(2006\)](#), which argues that the biggest gains in modeling occur when large-scale structure in the data is modeled using simple methods. Further performance increases tend to be incremental and provide diminishing returns. The advantages of the structured, sparse, fused sparse group lasso is that it is not a black box, but rather the estimated coefficients are readily interpretable, as it is an additive linear model. Interpretability and accuracy can be further enhanced by incorporating prior information. This may require more work upfront from the researcher, involving careful thinking about which prior information to include and how to best do so, but this extra work is advantageous if we care about gaining scientific insight into the system we seek to derive predictions from, and want to avoid results driven by confounding variables or spurious associations.

Nevertheless, fused sparse group lasso methods also have some disadvantages. The primary disadvantage of the current implementation of method is that it is computationally slow. Cross-validation for each (α, γ) scenario can take hours or days (depending on how large the data are), as compared to seconds for the `glmnet` package, which interfaces with Fortran code, a very fast language used for high performance computing. The fusion penalty in particular contributes to the increased computational burden of fused sparse group lasso, as the number of neighboring coefficient pairs for a three dimensional image is of the order p^3 . Thus, to bring the computational time within a reasonable range for the current fused sparse group lasso software, feature screening may be required, and use of a computing cluster to parallelize cross-validation steps may be essential. We suggest that researchers

first try fitting models with `glmnet`'s ridge, lasso, elastic net penalties to get an idea of how sparse the true model is likely to be (this can be inferred based on the optimal value for α in the elastic net penalty), and the likely optimal number of features (by trying several different sized sets of features). There are also fast packages for group lasso, such as `gglasso`, which can be used to explore different group structures. These preliminary steps will allow researchers to gain insight into the sparsity structure and what the best α and γ values are likely to be for fused sparse group lasso. As our software currently is set up, it requires the creation of a \mathbf{K} matrix as a separate step, and additional steps to transform data back and forth from the three dimensional image space into concatenated vectors. These steps can be tedious and time consuming, and could be further automated. We found that the structured penalties often outperformed the unstructured lasso, ridge, elastic net, or standard group lasso, in terms of prediction accuracy, but whether that degree of improvement is worth the additional hours it may take to implement the fused sparse group lasso penalty will depend on the research goals. For example, in the ABIDE application, the adaptive fused sparse group lasso had the best prediction accuracy, but it was only slightly better than the adaptive lasso, so apparently the additional structure did not help a large amount. This is consistent with the expectation of [Hand \(2006\)](#), discussed above. However, the brain map of estimated coefficients looked smoother and less scattered for the structured penalty, suggesting that if the goal is to generate an interpretable brain map, then it may be worth the additional effort. Finally, while the fused sparse group lasso methodology does produce individualized predictions, it assumes that the neurobiology is spatially the same across participants. This assumption is also invoked for other predictive methods such as support vector regression. Thus, such methods are not appropriate for making predictions based on pathology that varies spatially across individuals, such as white matter hyperintensities or brain lesions due to multiple sclerosis.

The results of this dissertation suggest several areas for future work. Regarding the applications to real data, for the ABIDE study (Section [2.4](#)), we note that the $n = 44$ test set participants were also used to derive the ICA resting state networks and determine which pairwise connectivity relationships associated with Social Responsiveness Scale scores in the [Cerliani et al. \(2015\)](#) study. Therefore, it may be somewhat circular to use the same

participants to validate the findings of the fused sparse group lasso application. Since [Cerliani et al. \(2015\)](#) used only a subset of participants in the ABIDE study, the estimated fused sparse group lasso models could be validated on other ABIDE participants. Additionally, it would be interesting to explore whether adding voxel-wise connectivity data for other seed regions, such as the posterior superior temporal sulcus ([Pelphrey et al. 2011](#)), improves prediction accuracy. For the Normal Aging study (Section 3.3), it may be beneficial to do the feature selection step in more systematic way, i.e., to search over several values for minimum cluster size and number of features to find optimal values for each modality. It would also be of interest to use a larger sample size, such as an ADNI sample, to train the age prediction model using the same methods, to see how much prediction accuracy might improve. A larger sample could also be used to further validate the models that we estimated, and assess association of the brain-PAD values with clinical and demographic variables such as cognitive scores or education. For both of the applications to real data and in the simulation study, cross-validation was used to select the optimal tuning parameters. Since cross-validation and model fitting is computationally intensive, this was not done, but it would be ideal to repeat the fold randomization some number of times (e.g. [Varikuti et al. \(2018\)](#) used 100 iterations of 10-fold cross-validation to tune each model) to better identify the minimum cross-validation error and estimate the optimal λ value. Additionally, it may be worth applying nested cross-validation to obtain predictions for each individual in the sample, particularly for the relatively small Normal Aging study.

Future simulation studies using the fused sparse group lasso estimator could explore adding spatial and group-structured correlation to the \mathbf{X} predictor matrix, to simulate data that more closely resembles neuroimaging data. Simulations could also explore the use of overlapping groups, as well as the adaptive penalties, which were found to work well on real data but were not studied in the simulations. Further methodological work could explore different fusion penalty weighting for edge voxels. Because they have fewer neighbors, edge voxels are more influenced by the neighbors they do have. This effect can be seen in Figure 2.9, where the 2×2 square touching the edge in the lower left is shrunk less than the other coefficients. This is particularly notable in the $\gamma = 0$ case. Downweighting the fusion penalties for edge voxels could help reduce the extra bias. Attention to proper weighting

of penalty terms is also crucial in the case of overlapping groups (Obozinski et al. 2011, Huo & Tseng 2017). Additional open methodological questions involve methods for feature screening, such as estimating optimal cluster size and number of features, as well as optimal voxel size. Work could be done comparing fused sparse group lasso regression with other prediction algorithms such as support vector regression, Gaussian process regression, random forest, and neural networks. The fused sparse group lasso penalty could be extended to use with other loss functions, such as logistic loss. Finally, potential future work regarding the software implementation of fused sparse group lasso includes submitting the `fsgl` R package to the Comprehensive R Archive Network (CRAN, R Core Team (2017), <https://cran.r-project.org/>), creating a vignette demonstrating its use, and increasing the computational efficiency of the optimization algorithm.

APPENDIX A

SIMULATION STUDY RESULTS

Table A1: Simulation results scenario 1A – Completely aggregated

α	γ	Frequency of minimum			Mean (SD) over 100 simulation iterations					Mean (SD) over 100 test obs.	
		Mean CVE	MSE($\hat{\beta}$)	MSE(\hat{y}_{test})	Mean CVE	SD CVE	MSE($\hat{\beta}$)	MSE(\hat{y}_{test})	Optimal λ	Bias ² ($\hat{y}_{\text{i test}}^*$)	Var($\hat{y}_{\text{i test}}^*$)
0.0	0.0	6	5	13	8.94 (5.57)	5.54 (8.29)	0.010 (0.008)	7.79 (3.42)	5.95 (3.08)	1.93 (2.35)	2.09 (0.65)
0.0	0.2	76	71	53	6.55 (2.03)	3.45 (1.80)	0.008 (0.013)	6.72 (3.34)	6.35 (2.97)	1.65 (2.05)	1.51 (0.84)
0.2	0.2	4	1	2	6.68 (2.13)	3.57 (1.94)	0.009 (0.022)	7.05 (5.19)	6.29 (2.97)	1.76 (2.19)	1.93 (1.37)
0.5	0.2	1	3	4	6.98 (2.37)	3.85 (2.36)	0.008 (0.006)	6.93 (3.14)	6.42 (2.79)	1.74 (2.17)	1.29 (0.42)
0.8	0.2	1	2	5	7.44 (2.79)	4.24 (3.12)	0.009 (0.017)	7.52 (6.36)	6.42 (3.03)	1.94 (2.44)	1.75 (0.90)
1.0	0.2	1	0	3	7.88 (3.25)	4.66 (3.94)	0.008 (0.005)	7.20 (2.78)	6.05 (3.14)	1.87 (2.36)	1.51 (0.41)
0.0	0.5	9	16	16	7.66 (2.53)	4.25 (2.62)	0.009 (0.012)	7.52 (5.40)	5.68 (3.01)	1.98 (2.47)	1.64 (0.77)
0.2	0.5	2	2	1	8.02 (2.81)	4.60 (3.05)	0.011 (0.019)	8.06 (6.72)	5.64 (3.02)	2.24 (2.76)	2.05 (1.06)
0.5	0.5	0	0	1	9.15 (3.80)	5.68 (4.57)	0.014 (0.033)	9.84 (16.66)	5.87 (3.30)	2.69 (3.37)	3.10 (2.16)
0.8	0.5	0	0	1	12.34 (8.02)	9.19 (11.02)	0.019 (0.038)	11.41 (14.99)	5.43 (3.43)	3.68 (4.59)	4.15 (2.50)
1.0	0.5	0	0	0	17.86 (12.93)	14.80 (15.66)	0.023 (0.037)	13.07 (16.54)	5.54 (3.75)	4.40 (5.59)	4.73 (2.74)
0.0	0.8	0	0	1	15.90 (7.80)	10.99 (7.98)	0.058 (0.088)	26.30 (35.33)	3.84 (3.34)	10.67 (13.46)	13.14 (10.68)
0.2	0.8	0	0	0	19.26 (10.35)	14.21 (10.66)	0.057 (0.093)	25.98 (36.81)	3.82 (3.45)	10.36 (13.04)	13.24 (10.78)
0.5	0.8	0	0	0	31.00 (17.92)	24.45 (16.92)	0.064 (0.090)	28.27 (33.80)	3.55 (3.69)	12.68 (16.19)	13.81 (10.41)
0.8	0.8	0	0	0	72.25 (31.30)	50.31 (24.60)	0.112 (0.106)	48.57 (44.95)	6.24 (9.94)	26.05 (33.08)	20.48 (12.58)
1.0	0.8	0	0	0	141.90 (32.96)	73.90 (32.59)	0.295 (0.172)	118.56 (72.32)	69.10 (162.39)	82.91 (106.96)	37.95 (22.28)
0.0	1.0	0	0	0	47.35 (17.60)	30.46 (14.63)	0.067 (0.042)	30.44 (17.95)	4.74 (5.46)	15.61 (20.22)	12.51 (4.46)
0.2	1.0	0	0	0	60.57 (19.61)	37.54 (16.93)	0.154 (0.114)	63.14 (45.42)	5.56 (8.35)	40.35 (52.79)	23.55 (13.62)
0.5	1.0	0	0	0	93.20 (21.19)	52.57 (22.68)	0.196 (0.088)	80.99 (40.17)	12.36 (15.73)	54.97 (71.33)	26.35 (8.74)
0.8	1.0	0	0	0	145.06 (29.35)	68.60 (29.24)	0.346 (0.083)	137.98 (42.60)	39.12 (47.25)	106.62 (139.48)	37.42 (9.39)
1.0	1.0	0	0	0	186.05 (40.96)	83.08 (38.29)	0.538 (0.045)	210.16 (38.85)	376.83 (409.69)	195.01 (256.55)	29.93 (7.02)

Notes:

- α controls balance between group lasso ($\alpha = 0$) and ℓ_1 lasso ($\alpha = 1$) penalty terms.
- γ controls balance between fusion ($\gamma = 0$) and sparsity ($\gamma = 1$) penalty terms.
- ‘Frequency of minimum’ columns give frequency of instances out of 100 of simulations yielding the minimum indicated error measure for each (α, γ) pair.
- Mean CVE (cross-validated error) is the minimum mean CVE over 5 cross-validation folds, used to select the optimal λ .
- MSE($\hat{\beta}$) is the mean squared error of the estimated coefficients for the model fitted to the entire training sample of $n = 50$.
- MSE(\hat{y}_{test}) is the mean squared error of the estimated response for a test sample of $n = 50$.
- Bias²($\hat{y}_{\text{i test}}^*$) and Var($\hat{y}_{\text{i test}}^*$) were calculated as follows: For each (α, γ) pair, the response for each observation in a test sample of $n = 100$ was calculated using each set of estimated coefficients from the 100 simulation iterations. The mean squared bias and variance of the predicted responses were calculated for each observation individually, and then the mean and standard deviations of these were calculated over the 100 observations.

**True Coefficients 1A:
Complete Group**



Table A2: Simulation results scenario 2B – Partially aggregated

α	γ	Frequency of minimum			Mean (SD) over 100 simulation iterations					Mean (SD) over 100 test obs.	
		Mean CVE	MSE($\hat{\beta}$)	MSE(\hat{y}_{test})	Mean CVE	SD CVE	MSE($\hat{\beta}$)	MSE(\hat{y}_{test})	Optimal λ	Bias ² ($\hat{y}_{\text{i test}}^*$)	Var($\hat{y}_{\text{i test}}^*$)
0.0	0.0	0	0	0	161.36 (38.23)	69.57 (25.79)	0.412 (0.100)	167.54 (57.01)	78.79 (182.06)	131.13 (202.89)	44.98 (15.58)
0.0	0.2	0	0	0	113.64 (24.05)	59.66 (26.82)	0.209 (0.115)	85.08 (49.99)	6.57 (8.37)	58.40 (88.40)	28.47 (12.76)
0.2	0.2	0	0	0	118.22 (24.93)	60.80 (26.99)	0.228 (0.115)	92.77 (51.14)	7.31 (9.69)	64.68 (97.75)	29.98 (12.59)
0.5	0.2	0	0	0	125.96 (27.07)	62.55 (27.38)	0.244 (0.110)	98.68 (48.34)	9.23 (10.92)	70.24 (106.28)	31.01 (11.59)
0.8	0.2	0	0	0	134.38 (29.66)	64.90 (27.22)	0.290 (0.112)	116.32 (49.83)	15.43 (28.79)	86.54 (130.15)	35.32 (12.20)
1.0	0.2	0	0	0	140.55 (30.60)	66.63 (27.36)	0.314 (0.114)	124.66 (51.72)	21.21 (42.36)	94.66 (141.73)	37.67 (13.44)
0.0	0.5	0	23	22	56.99 (22.74)	39.61 (21.30)	0.088 (0.088)	38.10 (38.52)	2.91 (4.28)	21.33 (31.40)	14.98 (10.04)
0.2	0.5	0	0	1	67.52 (23.86)	45.00 (22.91)	0.129 (0.116)	53.70 (47.89)	3.22 (5.26)	32.74 (47.75)	21.92 (14.92)
0.5	0.5	0	1	1	88.87 (24.52)	53.73 (26.47)	0.177 (0.120)	73.06 (48.94)	4.42 (6.96)	48.77 (71.19)	27.08 (15.57)
0.8	0.5	0	0	0	119.12 (25.94)	61.10 (28.37)	0.228 (0.120)	92.88 (51.75)	9.06 (12.88)	65.96 (96.20)	31.53 (13.15)
1.0	0.5	0	0	0	146.06 (31.46)	68.59 (31.76)	0.324 (0.118)	129.73 (53.16)	28.45 (101.18)	99.57 (145.03)	39.56 (14.04)
0.0	0.8	93	63	64	38.41 (15.34)	26.35 (15.05)	0.101 (0.118)	42.27 (44.70)	3.03 (3.95)	23.61 (33.58)	20.20 (16.04)
0.2	0.8	0	2	2	49.51 (18.34)	33.16 (17.59)	0.115 (0.115)	48.01 (44.68)	3.01 (4.66)	28.82 (41.05)	21.28 (14.90)
0.5	0.8	0	0	0	78.65 (21.80)	47.11 (21.52)	0.163 (0.118)	67.12 (46.56)	4.45 (7.00)	45.50 (64.89)	26.37 (15.66)
0.8	0.8	0	0	0	129.37 (28.36)	62.76 (28.92)	0.275 (0.115)	110.72 (47.70)	17.12 (32.92)	85.88 (123.17)	35.10 (13.10)
1.0	0.8	0	0	0	174.06 (39.29)	78.30 (35.14)	0.457 (0.089)	183.83 (50.87)	97.17 (162.28)	159.57 (230.09)	41.68 (12.49)
0.0	1.0	7	11	10	45.76 (15.16)	28.73 (14.82)	0.063 (0.041)	27.66 (16.91)	4.22 (4.79)	15.26 (21.43)	12.20 (4.40)
0.2	1.0	0	0	0	59.46 (17.51)	35.71 (17.10)	0.156 (0.129)	65.20 (53.71)	4.44 (5.57)	41.57 (58.72)	26.68 (17.32)
0.5	1.0	0	0	0	92.27 (22.19)	48.47 (21.58)	0.190 (0.098)	77.20 (40.61)	9.89 (11.65)	55.19 (78.11)	28.15 (10.45)
0.8	1.0	0	0	0	143.86 (30.55)	67.49 (29.89)	0.350 (0.082)	140.95 (40.99)	55.55 (145.28)	113.13 (160.89)	39.75 (10.90)
1.0	1.0	0	0	0	189.43 (41.04)	84.10 (35.06)	0.536 (0.040)	214.65 (41.14)	327.31 (377.91)	206.33 (297.76)	30.69 (7.50)

Notes:

- α controls balance between group lasso ($\alpha = 0$) and ℓ_1 lasso ($\alpha = 1$) penalty terms.
- γ controls balance between fusion ($\gamma = 0$) and sparsity ($\gamma = 1$) penalty terms.
- ‘Frequency of minimum’ columns give frequency of instances out of 100 of simulations yielding the minimum indicated error measure for each (α, γ) pair.
- Mean CVE (cross-validated error) is the minimum mean CVE over 5 cross-validation folds, which was used to select the optimal λ .
- MSE($\hat{\beta}$) is the mean squared error of the estimated coefficients for the model fitted to the entire training sample of $n = 50$.
- MSE(\hat{y}_{test}) is the mean squared error of the estimated response for a test sample of $n = 50$.
- Bias²($\hat{y}_{\text{i test}}^*$) and Var($\hat{y}_{\text{i test}}^*$) were calculated as follows: For each (α, γ) pair, the response for each observation in a test sample of $n = 100$ was calculated using each set of estimated coefficients from the 100 simulation iterations. The mean squared bias and variance of the predicted responses were calculated for each observation individually, and then the mean and standard deviations of these were calculated over the 100 observations.

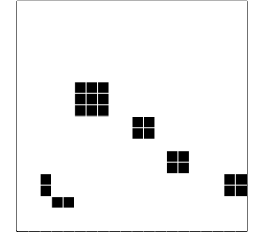
True Coefficients 2B:
Complete Group

Table A3: Simulation results scenario 3C – Completely distributed

α	γ	Frequency of minimum			Mean (SD) over 100 simulation iterations					Mean (SD) over 100 test obs.	
		Mean CVE	MSE($\hat{\beta}$)	MSE(\hat{y}_{test})	Mean CVE	SD CVE	MSE($\hat{\beta}$)	MSE(\hat{y}_{test})	Optimal λ	Bias ² ($\hat{y}_{\text{i test}}^*$)	Var($\hat{y}_{\text{i test}}^*$)
0.0	0.0	0	0	0	186.18 (38.07)	81.51 (30.54)	0.571 (0.060)	226.26 (53.30)	257.34 (330.13)	202.26 (339.84)	23.99 (9.36)
0.0	0.2	0	0	0	179.75 (36.07)	78.18 (31.74)	0.546 (0.030)	218.21 (54.85)	86.61 (174.00)	193.44 (325.28)	25.60 (6.94)
0.2	0.2	0	0	0	180.28 (36.15)	78.04 (31.97)	0.548 (0.028)	218.14 (53.68)	145.35 (269.96)	193.26 (324.96)	25.84 (6.91)
0.5	0.2	0	0	0	181.73 (36.47)	78.74 (32.52)	0.555 (0.029)	221.58 (55.77)	139.96 (243.00)	196.26 (329.17)	26.18 (6.87)
0.8	0.2	0	0	0	183.22 (36.54)	79.77 (32.22)	0.564 (0.031)	225.68 (57.14)	159.41 (251.41)	199.34 (331.90)	27.71 (7.17)
1.0	0.2	0	0	0	185.10 (37.10)	80.98 (31.04)	0.571 (0.033)	228.76 (59.70)	162.93 (262.06)	202.02 (336.57)	27.99 (7.42)
0.0	0.5	0	0	0	160.39 (26.65)	70.79 (29.82)	0.451 (0.058)	183.95 (50.80)	40.76 (142.82)	155.86 (261.08)	25.36 (5.65)
0.2	0.5	0	0	0	163.30 (28.21)	71.44 (29.99)	0.461 (0.056)	188.14 (51.68)	65.74 (185.52)	159.73 (266.73)	25.92 (6.00)
0.5	0.5	0	0	0	170.64 (32.13)	73.87 (30.19)	0.488 (0.049)	197.49 (52.89)	48.66 (111.00)	169.75 (283.79)	26.10 (5.70)
0.8	0.5	0	0	0	179.83 (36.16)	77.06 (31.98)	0.527 (0.041)	211.97 (55.64)	94.74 (176.03)	187.21 (311.47)	24.64 (5.42)
1.0	0.5	0	0	0	186.83 (37.65)	80.09 (31.39)	0.559 (0.033)	223.33 (57.10)	202.68 (316.35)	204.64 (341.07)	22.36 (5.31)
0.0	0.8	0	0	0	111.58 (23.99)	57.98 (24.72)	0.313 (0.113)	129.59 (56.85)	19.17 (100.26)	98.94 (165.22)	26.51 (9.93)
0.2	0.8	0	0	0	121.00 (23.23)	60.04 (25.52)	0.320 (0.092)	132.44 (51.23)	21.33 (100.50)	101.30 (169.70)	27.07 (8.34)
0.5	0.8	0	0	0	140.00 (23.37)	64.36 (27.52)	0.374 (0.085)	152.60 (48.86)	33.04 (105.68)	120.12 (201.28)	30.05 (8.78)
0.8	0.8	0	0	0	165.60 (29.98)	71.43 (30.36)	0.461 (0.064)	186.33 (51.75)	62.33 (109.93)	153.97 (255.86)	30.33 (7.86)
1.0	0.8	0	0	0	183.26 (37.49)	78.29 (31.71)	0.547 (0.040)	220.87 (55.85)	143.25 (151.26)	194.99 (325.58)	26.05 (6.40)
0.0	1.0	100	97	97	52.43 (20.49)	34.19 (17.19)	0.075 (0.049)	34.30 (21.80)	5.08 (6.53)	17.11 (28.67)	13.01 (5.45)
0.2	1.0	0	3	3	65.85 (22.59)	40.02 (18.90)	0.156 (0.130)	64.80 (51.88)	6.30 (8.77)	37.80 (62.75)	23.92 (16.18)
0.5	1.0	0	0	0	96.66 (23.95)	50.94 (21.39)	0.209 (0.106)	86.93 (46.70)	11.53 (15.86)	55.23 (92.19)	28.21 (12.10)
0.8	1.0	0	0	0	142.42 (24.75)	65.24 (27.57)	0.365 (0.082)	149.83 (44.95)	54.64 (110.55)	112.07 (186.71)	34.32 (10.64)
1.0	1.0	0	0	0	181.07 (37.81)	79.12 (32.55)	0.543 (0.038)	219.54 (55.27)	274.52 (333.78)	190.52 (317.56)	30.71 (7.40)

Notes:

- α controls balance between group lasso ($\alpha = 0$) and ℓ_1 lasso ($\alpha = 1$) penalty terms.
- γ controls balance between fusion ($\gamma = 0$) and sparsity ($\gamma = 1$) penalty terms.
- ‘Frequency of minimum’ columns give frequency of instances out of 100 of simulations yielding the minimum indicated error measure for each (α, γ) pair.
- Mean CVE (cross-validated error) is the minimum mean CVE over 5 cross-validation folds, which was used to select the optimal λ .
- MSE($\hat{\beta}$) is the mean squared error of the estimated coefficients for the model fitted to the entire training sample of $n = 50$.
- MSE(\hat{y}_{test}) is the mean squared error of the estimated response for a test sample of $n = 50$.
- Bias²($\hat{y}_{\text{i test}}^*$) and Var($\hat{y}_{\text{i test}}^*$) were calculated as follows: For each (α, γ) pair, the response for each observation in a test sample of $n = 100$ was calculated using each set of estimated coefficients from the 100 simulation iterations. The mean squared bias and variance of the predicted responses were calculated for each observation individually, and then the mean and standard deviations of these were calculated over the 100 observations.

**True Coefficients 3C:
Complete Group**

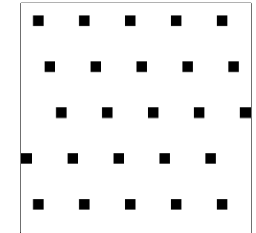


Table A4: Simulation results scenario 4A – Sparse completely aggregated

α	γ	Frequency of minimum			Mean (SD) over 100 simulation iterations					Mean (SD) over 100 test obs.	
		Mean CVE	MSE($\hat{\beta}$)	MSE($\hat{\mathbf{y}}_{\text{test}}$)	Mean CVE	SD CVE	MSE($\hat{\beta}$)	MSE($\hat{\mathbf{y}}_{\text{test}}$)	Optimal λ	Bias ² ($\hat{y}_{i \text{ test}}^*$)	Var($\hat{y}_{i \text{ test}}^*$)
0.0	0.0	0	2	7	12.23 (7.20)	7.87 (7.47)	0.013 (0.011)	9.03 (4.82)	5.33 (2.67)	2.14 (3.68)	2.52 (0.88)
0.0	0.2	38	33	23	8.32 (3.45)	5.04 (3.97)	0.010 (0.012)	7.81 (4.91)	5.67 (2.77)	1.88 (3.18)	1.77 (0.79)
0.2	0.2	12	11	9	8.37 (3.53)	5.05 (4.02)	0.010 (0.012)	7.95 (5.37)	5.62 (2.80)	1.89 (3.20)	1.83 (0.80)
0.5	0.2	8	14	9	8.52 (3.67)	5.11 (4.18)	0.012 (0.026)	8.94 (11.15)	5.75 (2.82)	2.11 (3.56)	2.37 (1.56)
0.8	0.2	8	8	5	8.84 (3.98)	5.29 (4.38)	0.011 (0.021)	8.47 (9.11)	5.84 (2.97)	2.03 (3.39)	2.15 (1.18)
1.0	0.2	2	3	11	9.12 (4.25)	5.49 (4.79)	0.010 (0.007)	7.77 (3.13)	5.58 (3.05)	1.90 (3.14)	1.77 (0.57)
0.0	0.5	9	8	11	8.79 (3.22)	5.38 (3.94)	0.013 (0.026)	9.15 (11.37)	5.20 (2.97)	2.37 (3.87)	2.49 (1.64)
0.2	0.5	9	8	10	8.73 (3.31)	5.39 (4.07)	0.013 (0.024)	9.30 (11.08)	5.33 (3.01)	2.38 (3.88)	2.51 (1.67)
0.5	0.5	9	7	7	9.02 (3.75)	5.67 (4.81)	0.016 (0.028)	10.23 (11.72)	5.47 (2.88)	2.76 (4.51)	3.21 (2.58)
0.8	0.5	1	4	6	10.09 (4.81)	6.61 (6.05)	0.014 (0.023)	9.57 (10.40)	5.80 (3.12)	2.59 (4.17)	2.65 (1.54)
1.0	0.5	4	2	2	12.34 (7.18)	8.79 (9.16)	0.014 (0.010)	9.39 (4.47)	5.82 (3.87)	2.80 (4.47)	2.51 (0.80)
0.0	0.8	0	0	0	15.21 (6.11)	9.74 (6.95)	0.038 (0.054)	18.73 (21.73)	4.53 (3.52)	6.64 (10.63)	7.56 (6.41)
0.2	0.8	0	0	0	15.02 (6.42)	9.78 (7.49)	0.036 (0.054)	17.78 (21.21)	4.70 (3.67)	6.16 (9.83)	7.26 (6.10)
0.5	0.8	0	0	0	16.42 (8.14)	11.26 (9.63)	0.025 (0.036)	14.05 (15.74)	5.32 (4.22)	4.63 (7.35)	4.83 (2.97)
0.8	0.8	0	0	0	25.02 (14.00)	18.42 (14.67)	0.034 (0.044)	17.36 (17.36)	6.09 (5.18)	6.54 (10.30)	6.49 (4.51)
1.0	0.8	0	0	0	54.74 (23.08)	38.92 (23.38)	0.070 (0.073)	31.33 (31.13)	19.64 (99.65)	13.73 (22.20)	12.18 (8.35)
0.0	1.0	0	0	0	32.87 (10.61)	19.86 (11.04)	0.049 (0.028)	23.43 (12.33)	5.06 (5.72)	10.08 (15.68)	8.82 (2.91)
0.2	1.0	0	0	0	34.33 (11.52)	21.08 (11.64)	0.078 (0.064)	34.09 (25.99)	4.80 (5.85)	16.70 (26.39)	12.39 (7.50)
0.5	1.0	0	0	0	42.24 (13.71)	25.65 (13.13)	0.070 (0.049)	30.99 (19.56)	7.06 (8.01)	14.75 (23.58)	11.47 (5.42)
0.8	1.0	0	0	0	69.08 (15.29)	37.38 (16.70)	0.127 (0.068)	54.56 (30.75)	27.79 (100.97)	29.18 (47.24)	17.96 (7.64)
1.0	1.0	0	0	0	106.45 (20.56)	48.07 (21.44)	0.283 (0.048)	112.35 (28.24)	174.63 (293.11)	79.35 (128.34)	25.01 (6.42)

Notes:

- α controls balance between group lasso ($\alpha = 0$) and ℓ_1 lasso ($\alpha = 1$) penalty terms.
- γ controls balance between fusion ($\gamma = 0$) and sparsity ($\gamma = 1$) penalty terms.
- ‘Frequency of minimum’ columns give frequency of instances out of 100 of simulations yielding the minimum indicated error measure for each (α, γ) pair.
- Mean CVE (cross-validated error) is the minimum mean CVE over 5 cross-validation folds, which was used to select the optimal λ .
- MSE($\hat{\beta}$) is the mean squared error of the estimated coefficients for the model fitted to the entire training sample of $n = 50$.
- MSE($\hat{\mathbf{y}}_{\text{test}}$) is the mean squared error of the estimated response for a test sample of $n = 50$.
- Bias²($\hat{y}_{i \text{ test}}^*$) and Var($\hat{y}_{i \text{ test}}^*$) were calculated as follows: For each (α, γ) pair, the response for each observation in a test sample of $n = 100$ was calculated using each set of estimated coefficients from the 100 simulation iterations. The mean squared bias and variance of the predicted responses were calculated for each observation individually, and then the mean and standard deviations of these were calculated over the 100 observations.

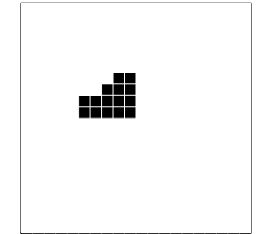
True Coefficients 4A:
Sparse Group

Table A5: Simulation results scenario 5B – Sparse partially aggregated

α	γ	Frequency of minimum			Mean (SD) over 100 simulation iterations					Mean (SD) over 100 test obs.	
		Mean CVE	MSE($\hat{\beta}$)	MSE(\hat{y}_{test})	Mean CVE	SD CVE	MSE($\hat{\beta}$)	MSE(\hat{y}_{test})	Optimal λ	Bias ² ($\hat{y}_{\text{i test}}^*$)	Var($\hat{y}_{\text{i test}}^*$)
0.0	0.0	0	0	0	60.74 (18.60)	38.84 (16.15)	0.081 (0.072)	34.43 (28.12)	11.66 (44.69)	17.92 (23.46)	14.90 (7.58)
0.0	0.2	0	7	6	24.71 (13.21)	18.45 (12.05)	0.031 (0.037)	15.36 (13.71)	3.85 (3.29)	7.13 (9.11)	5.51 (2.68)
0.2	0.2	1	0	2	25.92 (13.67)	19.27 (12.27)	0.030 (0.038)	15.41 (15.01)	3.73 (3.36)	6.97 (8.91)	5.66 (3.09)
0.5	0.2	0	0	0	28.54 (14.53)	21.27 (12.75)	0.038 (0.047)	18.73 (20.58)	3.57 (4.02)	8.35 (10.72)	7.58 (4.50)
0.8	0.2	0	0	1	32.64 (15.26)	24.24 (13.61)	0.050 (0.058)	23.24 (24.56)	4.09 (5.82)	10.96 (13.93)	9.98 (5.95)
1.0	0.2	0	0	0	36.81 (15.72)	27.36 (14.23)	0.042 (0.043)	19.95 (17.93)	4.26 (5.15)	9.40 (12.06)	7.98 (4.04)
0.0	0.5	87	74	61	14.56 (6.62)	9.83 (6.12)	0.021 (0.032)	12.36 (14.26)	3.89 (2.96)	5.12 (6.57)	3.94 (2.39)
0.2	0.5	8	8	11	15.38 (7.25)	10.67 (6.86)	0.017 (0.016)	10.45 (7.47)	3.91 (3.00)	4.51 (5.79)	2.88 (1.30)
0.5	0.5	1	6	7	18.32 (9.19)	13.45 (9.18)	0.033 (0.051)	16.23 (20.11)	4.04 (3.06)	7.29 (9.46)	7.05 (5.14)
0.8	0.5	0	1	2	25.99 (13.22)	19.58 (13.43)	0.043 (0.052)	20.52 (21.99)	3.89 (3.89)	9.89 (12.72)	8.56 (5.60)
1.0	0.5	0	0	0	38.40 (16.84)	28.04 (15.05)	0.054 (0.058)	25.11 (24.34)	4.70 (6.05)	12.71 (16.37)	10.56 (5.90)
0.0	0.8	2	4	5	18.94 (7.27)	12.35 (6.54)	0.046 (0.059)	21.12 (22.36)	3.31 (3.29)	11.08 (14.21)	9.64 (7.29)
0.2	0.8	1	0	4	19.60 (7.81)	13.11 (7.20)	0.049 (0.065)	22.70 (25.69)	3.32 (3.47)	11.59 (14.84)	10.21 (8.00)
0.5	0.8	0	0	1	23.72 (10.21)	16.48 (9.79)	0.045 (0.057)	20.91 (21.97)	3.81 (4.15)	10.87 (13.89)	9.13 (6.20)
0.8	0.8	0	0	0	40.48 (16.25)	27.24 (14.98)	0.054 (0.052)	25.03 (22.14)	5.83 (6.82)	13.90 (17.85)	9.86 (5.74)
1.0	0.8	0	0	0	77.15 (20.95)	40.52 (18.02)	0.134 (0.094)	56.78 (39.95)	25.28 (45.77)	37.83 (49.00)	20.71 (12.13)
0.0	1.0	0	0	0	33.75 (9.65)	20.27 (9.30)	0.047 (0.027)	21.80 (12.18)	5.00 (5.13)	12.38 (15.65)	8.36 (3.00)
0.2	1.0	0	0	0	35.27 (10.14)	21.65 (10.06)	0.070 (0.059)	30.93 (25.13)	4.39 (5.31)	18.67 (23.87)	12.18 (6.50)
0.5	1.0	0	0	0	43.42 (12.12)	26.68 (12.58)	0.077 (0.063)	33.82 (26.18)	6.12 (7.23)	21.20 (26.92)	13.29 (7.28)
0.8	1.0	0	0	0	70.41 (16.46)	36.26 (15.87)	0.129 (0.068)	54.71 (31.25)	27.25 (99.98)	38.48 (49.36)	18.39 (7.99)
1.0	1.0	0	0	0	102.05 (19.98)	43.72 (19.03)	0.290 (0.051)	118.21 (29.39)	144.70 (261.19)	101.53 (130.82)	27.52 (8.09)

Notes:

- α controls balance between group lasso ($\alpha = 0$) and ℓ_1 lasso ($\alpha = 1$) penalty terms.
- γ controls balance between fusion ($\gamma = 0$) and sparsity ($\gamma = 1$) penalty terms.
- ‘Frequency of minimum’ columns give frequency of instances out of 100 of simulations yielding the minimum indicated error measure for each (α, γ) pair.
- Mean CVE (cross-validated error) is the minimum mean CVE over 5 cross-validation folds, which was used to select the optimal λ .
- MSE($\hat{\beta}$) is the mean squared error of the estimated coefficients for the model fitted to the entire training sample of $n = 50$.
- MSE(\hat{y}_{test}) is the mean squared error of the estimated response for a test sample of $n = 50$.
- Bias²($\hat{y}_{\text{i test}}^*$) and Var($\hat{y}_{\text{i test}}^*$) were calculated as follows: For each (α, γ) pair, the response for each observation in a test sample of $n = 100$ was calculated using each set of estimated coefficients from the 100 simulation iterations. The mean squared bias and variance of the predicted responses were calculated for each observation individually, and then the mean and standard deviations of these were calculated over the 100 observations.

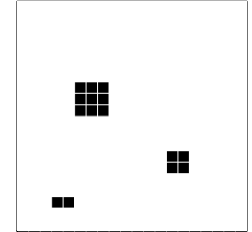
True Coefficients 5B:
Sparse Group

Table A6: Simulation results scenario 6C – Sparse completely distributed

α	γ	Frequency of minimum			Mean (SD) over 100 simulation iterations					Mean (SD) over 100 test obs.	
		Mean CVE	MSE($\hat{\beta}$)	MSE(\hat{y}_{test})	Mean CVE	SD CVE	MSE($\hat{\beta}$)	MSE(\hat{y}_{test})	Optimal λ	Bias ² ($\hat{y}_{\text{i test}}^*$)	Var($\hat{y}_{\text{i test}}^*$)
0.0	0.0	0	0	0	114.11 (22.85)	48.67 (19.49)	0.346 (0.030)	137.31 (30.12)	227.33 (327.26)	133.51 (202.74)	15.02 (5.65)
0.0	0.2	0	0	0	106.42 (20.47)	45.42 (18.89)	0.314 (0.026)	125.47 (27.25)	48.10 (106.77)	117.40 (180.09)	16.73 (4.82)
0.2	0.2	0	0	0	106.70 (20.55)	45.49 (18.96)	0.316 (0.026)	125.97 (26.69)	119.30 (264.32)	118.29 (181.42)	16.78 (4.76)
0.5	0.2	0	0	0	107.73 (20.83)	45.43 (18.96)	0.320 (0.024)	127.27 (26.33)	100.34 (204.82)	119.21 (183.36)	17.14 (4.82)
0.8	0.2	0	0	0	109.24 (21.23)	46.65 (20.24)	0.326 (0.022)	130.24 (26.47)	114.59 (220.97)	121.97 (187.39)	17.05 (4.86)
1.0	0.2	0	0	0	110.24 (21.33)	46.98 (20.48)	0.331 (0.022)	132.43 (27.80)	142.40 (246.78)	124.97 (192.42)	16.31 (4.76)
0.0	0.5	0	0	0	88.49 (16.21)	41.65 (16.50)	0.237 (0.049)	97.52 (29.58)	20.62 (101.95)	83.78 (129.96)	17.29 (4.77)
0.2	0.5	0	0	0	90.00 (16.73)	41.72 (16.47)	0.240 (0.045)	99.07 (29.60)	22.52 (84.51)	84.68 (131.69)	17.67 (4.92)
0.5	0.5	0	0	0	94.13 (18.25)	42.84 (17.81)	0.262 (0.040)	106.71 (28.62)	30.94 (106.01)	93.03 (144.94)	18.00 (4.74)
0.8	0.5	0	0	0	101.05 (20.87)	44.43 (19.01)	0.288 (0.033)	117.69 (29.07)	56.95 (144.44)	103.71 (160.88)	18.82 (5.06)
1.0	0.5	0	0	0	105.72 (22.16)	45.74 (19.36)	0.316 (0.028)	127.86 (29.32)	135.74 (278.01)	113.37 (175.64)	20.15 (5.62)
0.0	0.8	0	1	1	55.42 (12.68)	32.90 (13.89)	0.128 (0.076)	54.93 (34.12)	5.08 (6.90)	38.94 (60.79)	15.38 (8.56)
0.2	0.8	0	0	0	59.46 (12.62)	34.04 (14.98)	0.146 (0.083)	61.82 (35.45)	7.28 (12.95)	45.02 (69.95)	16.99 (9.68)
0.5	0.8	0	0	0	69.60 (13.26)	36.45 (16.20)	0.166 (0.065)	70.64 (31.68)	13.11 (33.77)	51.99 (81.43)	17.36 (6.58)
0.8	0.8	0	0	0	87.08 (17.96)	40.70 (17.92)	0.229 (0.052)	95.89 (31.57)	26.82 (30.56)	76.15 (118.67)	19.39 (5.77)
1.0	0.8	0	0	0	103.03 (22.38)	46.77 (18.43)	0.301 (0.034)	122.19 (29.10)	134.00 (235.90)	107.22 (167.16)	19.25 (5.42)
0.0	1.0	85	46	46	33.97 (10.56)	21.11 (9.57)	0.058 (0.037)	27.31 (14.85)	5.84 (5.54)	14.33 (22.74)	9.60 (4.35)
0.2	1.0	15	43	42	35.44 (11.07)	22.21 (9.97)	0.081 (0.075)	36.55 (32.01)	5.83 (6.00)	20.11 (31.56)	13.97 (10.13)
0.5	1.0	0	10	11	43.39 (12.19)	26.67 (11.73)	0.088 (0.070)	39.44 (29.69)	6.97 (7.63)	22.39 (35.05)	14.11 (8.47)
0.8	1.0	0	0	0	67.77 (14.00)	35.34 (16.81)	0.143 (0.068)	62.04 (31.64)	19.32 (25.39)	40.18 (62.61)	18.71 (8.20)
1.0	1.0	0	0	0	100.76 (21.93)	45.50 (17.48)	0.290 (0.040)	119.29 (30.49)	177.75 (284.93)	98.77 (153.94)	21.48 (6.27)

Notes:

- α controls balance between group lasso ($\alpha = 0$) and ℓ_1 lasso ($\alpha = 1$) penalty terms.
- γ controls balance between fusion ($\gamma = 0$) and sparsity ($\gamma = 1$) penalty terms.
- ‘Frequency of minimum’ columns give frequency of instances out of 100 of simulations yielding the minimum indicated error measure for each (α, γ) pair.
- Mean CVE (cross-validated error) is the minimum mean CVE over 5 cross-validation folds, which was used to select the optimal λ .
- $\text{MSE}(\hat{\beta})$ is the mean squared error of the estimated coefficients for the model fitted to the entire training sample of $n = 50$.
- $\text{MSE}(\hat{y}_{\text{test}})$ is the mean squared error of the estimated response for a test sample of $n = 50$.
- $\text{Bias}^2(\hat{y}_{\text{i test}}^*)$ and $\text{Var}(\hat{y}_{\text{i test}}^*)$ were calculated as follows: For each (α, γ) pair, the response for each observation in a test sample of $n = 100$ was calculated using each set of estimated coefficients from the 100 simulation iterations. The mean squared bias and variance of the predicted responses were calculated for each observation individually, and then the mean and standard deviations of these were calculated over the 100 observations.

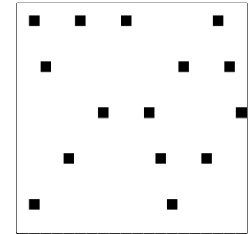
True Coefficients 6C:
Sparse Group

Table A7: Simulation results scenario 7B – Extra sparse partially aggregated

α	γ	Frequency of minimum			Mean (SD) over 100 simulation iterations					Mean (SD) over 100 test obs.	
		Mean CVE	MSE($\hat{\beta}$)	MSE($\hat{\mathbf{y}}_{\text{test}}$)	Mean CVE	SD CVE	MSE($\hat{\beta}$)	MSE($\hat{\mathbf{y}}_{\text{test}}$)	Optimal λ	Bias ² ($\hat{y}_{i \text{ test}}^*$)	Var($\hat{y}_{i \text{ test}}^*$)
0.0	0.0	0	0	0	21.39 (7.72)	14.64 (7.10)	0.031 (0.027)	15.50 (11.15)	15.59 (100.02)	4.22 (5.27)	5.49 (2.16)
0.0	0.2	0	1	0	11.32 (4.64)	7.02 (4.27)	0.014 (0.013)	9.15 (5.33)	4.54 (2.69)	2.20 (2.71)	2.14 (0.79)
0.2	0.2	1	0	0	11.25 (4.64)	6.93 (4.19)	0.014 (0.014)	9.25 (5.78)	4.46 (2.72)	2.19 (2.71)	2.25 (0.82)
0.5	0.2	1	1	2	11.34 (4.78)	6.98 (4.26)	0.015 (0.014)	9.60 (6.17)	4.65 (2.94)	2.26 (2.78)	2.46 (0.88)
0.8	0.2	1	0	0	11.70 (5.04)	7.16 (4.52)	0.016 (0.015)	9.86 (6.72)	4.42 (3.18)	2.27 (2.79)	2.75 (0.99)
1.0	0.2	0	0	0	12.16 (5.30)	7.31 (4.63)	0.017 (0.016)	10.31 (7.24)	4.56 (3.40)	2.36 (2.91)	2.97 (1.05)
0.0	0.5	0	5	4	9.21 (3.09)	5.47 (3.30)	0.012 (0.012)	8.45 (4.74)	4.73 (3.00)	1.97 (2.37)	1.72 (0.72)
0.2	0.5	3	8	8	8.76 (2.94)	5.11 (3.07)	0.012 (0.010)	8.32 (4.62)	4.93 (3.04)	1.87 (2.26)	1.68 (0.68)
0.5	0.5	22	19	12	8.45 (2.88)	4.80 (2.91)	0.011 (0.011)	7.89 (5.35)	5.27 (3.08)	1.64 (1.97)	1.68 (0.72)
0.8	0.5	15	18	18	8.69 (3.13)	4.99 (2.97)	0.011 (0.010)	7.92 (4.58)	6.25 (3.49)	1.65 (1.96)	1.77 (0.70)
1.0	0.5	5	7	9	9.59 (3.74)	5.60 (3.61)	0.012 (0.010)	8.31 (4.68)	6.39 (3.81)	1.72 (2.05)	2.08 (0.71)
0.0	0.8	1	0	0	11.58 (3.43)	6.92 (3.90)	0.019 (0.015)	11.11 (5.83)	4.41 (3.54)	3.07 (3.61)	2.67 (0.96)
0.2	0.8	0	0	3	10.28 (3.18)	6.14 (3.49)	0.016 (0.015)	10.06 (6.21)	5.04 (3.82)	2.55 (2.99)	2.38 (0.89)
0.5	0.8	9	9	8	8.87 (2.92)	5.22 (3.04)	0.015 (0.017)	9.36 (6.85)	6.07 (4.16)	2.14 (2.53)	2.29 (1.02)
0.8	0.8	34	31	32	8.46 (3.02)	4.98 (3.32)	0.011 (0.011)	8.21 (5.33)	7.68 (4.65)	1.67 (1.98)	1.87 (0.74)
1.0	0.8	4	0	2	10.04 (4.01)	5.96 (3.80)	0.013 (0.010)	8.54 (4.59)	8.80 (5.41)	1.76 (2.11)	2.28 (0.75)
0.0	1.0	0	0	0	16.85 (4.01)	9.27 (4.88)	0.027 (0.011)	14.07 (5.50)	6.22 (5.43)	4.37 (5.13)	4.01 (1.23)
0.2	1.0	0	0	0	14.63 (3.80)	8.29 (4.41)	0.026 (0.017)	13.58 (6.35)	6.05 (5.20)	4.23 (4.94)	3.69 (1.16)
0.5	1.0	0	0	2	11.53 (3.51)	6.69 (3.62)	0.016 (0.009)	9.80 (4.24)	6.59 (5.15)	2.45 (2.87)	2.46 (0.80)
0.8	1.0	4	1	0	10.01 (3.56)	6.00 (3.47)	0.013 (0.009)	8.66 (4.16)	9.40 (6.85)	1.93 (2.25)	2.14 (0.78)
1.0	1.0	0	0	0	14.43 (5.59)	9.23 (5.71)	0.020 (0.016)	11.51 (7.50)	14.31 (12.13)	2.80 (3.28)	3.60 (1.40)

Notes:

- α controls balance between group lasso ($\alpha = 0$) and ℓ_1 lasso ($\alpha = 1$) penalty terms.
- γ controls balance between fusion ($\gamma = 0$) and sparsity ($\gamma = 1$) penalty terms.
- ‘Frequency of minimum’ columns give frequency of instances out of 100 of simulations yielding the minimum indicated error measure for each (α, γ) pair.
- Mean CVE (cross-validated error) is the minimum mean CVE over 5 cross-validation folds, which was used to select the optimal λ .
- $\text{MSE}(\hat{\beta})$ is the mean squared error of the estimated coefficients for the model fitted to the entire training sample of $n = 50$.
- $\text{MSE}(\hat{\mathbf{y}}_{\text{test}})$ is the mean squared error of the estimated response for a test sample of $n = 50$.
- $\text{Bias}^2(\hat{y}_{i \text{ test}}^*)$ and $\text{Var}(\hat{y}_{i \text{ test}}^*)$ were calculated as follows: For each (α, γ) pair, the response for each observation in a test sample of $n = 100$ was calculated using each set of estimated coefficients from the 100 simulation iterations. The mean squared bias and variance of the predicted responses were calculated for each observation individually, and then the mean and standard deviations of these were calculated over the 100 observations.

**True Coefficients 7B:
Extra Sparse Group**

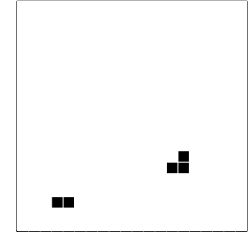


Table A8: Simulation results scenario 8B – Misspecified partially aggregated

α	γ	Frequency of minimum			Mean (SD) over 100 simulation iterations					Mean (SD) over 100 test obs.	
		Mean CVE	MSE($\hat{\beta}$)	MSE(\hat{y}_{test})	Mean CVE	SD CVE	MSE($\hat{\beta}$)	MSE(\hat{y}_{test})	Optimal λ	Bias ² ($\hat{y}_{\text{i test}}^*$)	Var($\hat{y}_{\text{i test}}^*$)
0.0	0.0	10	1	2	151.50 (34.20)	65.97 (26.18)	0.398 (0.096)	156.39 (46.87)	49.86 (118.57)	126.56 (177.64)	38.71 (13.11)
0.0	0.2	4	0	3	142.43 (28.98)	63.77 (25.98)	0.357 (0.082)	140.30 (41.22)	15.97 (30.86)	113.70 (167.41)	32.32 (9.21)
0.2	0.2	4	2	1	140.45 (28.87)	63.39 (25.84)	0.348 (0.087)	137.27 (41.70)	29.16 (101.83)	110.26 (162.61)	32.15 (9.51)
0.5	0.2	4	4	6	138.00 (28.85)	63.85 (26.03)	0.333 (0.092)	131.93 (42.26)	25.00 (76.71)	104.08 (154.64)	32.38 (9.60)
0.8	0.2	4	13	3	135.65 (29.47)	63.57 (26.44)	0.324 (0.102)	128.45 (45.06)	42.49 (144.88)	100.22 (149.36)	32.60 (10.29)
1.0	0.2	34	34	33	134.09 (29.94)	64.33 (27.88)	0.313 (0.103)	123.58 (44.67)	27.30 (76.44)	95.64 (142.71)	32.65 (10.18)
0.0	0.5	2	2	2	152.96 (28.96)	66.43 (28.43)	0.394 (0.056)	153.40 (34.31)	9.00 (19.43)	128.86 (195.84)	28.78 (5.87)
0.2	0.5	1	1	1	150.01 (28.38)	66.00 (28.78)	0.381 (0.064)	148.79 (35.11)	15.31 (54.18)	123.31 (186.70)	29.12 (6.57)
0.5	0.5	0	2	3	145.38 (28.18)	65.84 (29.93)	0.359 (0.074)	140.95 (36.72)	12.78 (25.78)	113.80 (173.23)	29.90 (6.99)
0.8	0.5	9	7	8	141.57 (28.70)	65.28 (30.36)	0.330 (0.094)	129.48 (42.97)	15.98 (28.68)	99.90 (152.01)	32.08 (9.23)
1.0	0.5	13	33	34	141.65 (30.88)	66.16 (29.11)	0.321 (0.109)	125.84 (46.55)	23.17 (34.29)	95.93 (146.64)	33.59 (10.50)
0.0	0.8	1	1	1	169.75 (35.38)	71.04 (27.84)	0.455 (0.048)	175.19 (38.10)	31.86 (111.09)	153.02 (234.84)	26.48 (5.53)
0.2	0.8	1	0	1	167.49 (34.50)	70.38 (28.86)	0.445 (0.051)	172.08 (38.26)	18.44 (35.66)	147.54 (226.18)	27.36 (5.95)
0.5	0.8	0	0	1	164.91 (33.28)	70.34 (31.17)	0.432 (0.060)	166.88 (36.26)	20.55 (37.99)	139.73 (213.46)	30.46 (6.64)
0.8	0.8	2	0	0	164.13 (34.34)	72.87 (31.52)	0.422 (0.073)	163.97 (39.91)	32.03 (48.05)	132.80 (202.47)	33.80 (8.08)
1.0	0.8	3	0	0	170.82 (35.91)	74.67 (32.96)	0.454 (0.098)	176.08 (49.83)	93.30 (137.74)	139.90 (214.04)	38.19 (11.20)
0.0	1.0	2	0	0	181.79 (40.05)	75.84 (30.05)	0.517 (0.040)	198.43 (42.48)	206.69 (375.26)	177.13 (270.24)	25.31 (5.95)
0.2	1.0	1	0	0	180.80 (39.62)	76.67 (29.93)	0.510 (0.046)	197.24 (42.05)	83.45 (124.26)	176.52 (270.56)	24.01 (5.80)
0.5	1.0	1	0	1	179.73 (39.58)	76.43 (30.09)	0.502 (0.050)	193.35 (39.75)	86.22 (177.72)	169.61 (258.94)	27.29 (6.54)
0.8	1.0	1	0	0	180.22 (39.41)	76.71 (30.79)	0.498 (0.050)	192.15 (41.71)	116.20 (217.00)	165.13 (250.72)	29.82 (7.26)
1.0	1.0	3	0	0	184.40 (40.95)	80.37 (32.84)	0.532 (0.046)	205.88 (43.85)	320.65 (391.41)	175.88 (266.54)	32.77 (8.43)

Notes:

- α controls balance between group lasso ($\alpha = 0$) and ℓ_1 lasso ($\alpha = 1$) penalty terms.
- γ controls balance between fusion ($\gamma = 0$) and sparsity ($\gamma = 1$) penalty terms.
- ‘Frequency of minimum’ columns give frequency of instances out of 100 of simulations yielding the minimum indicated error measure for each (α, γ) pair.
- Mean CVE (cross-validated error) is the minimum mean CVE over 5 cross-validation folds, which was used to select the optimal λ .
- MSE($\hat{\beta}$) is the mean squared error of the estimated coefficients for the model fitted to the entire training sample of $n = 50$.
- MSE(\hat{y}_{test}) is the mean squared error of the estimated response for a test sample of $n = 50$.
- Bias²($\hat{y}_{\text{i test}}^*$) and Var($\hat{y}_{\text{i test}}^*$) were calculated as follows: For each (α, γ) pair, the response for each observation in a test sample of $n = 100$ was calculated using each set of estimated coefficients from the 100 simulation iterations. The mean squared bias and variance of the predicted responses were calculated for each observation individually, and then the mean and standard deviations of these were calculated over the 100 observations.

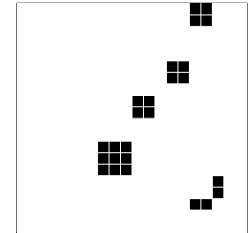
True Coefficients 8B:
Misspecified Group

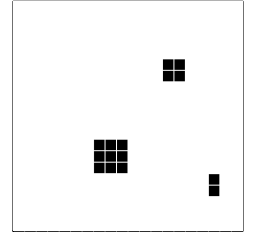
Table A9: Simulation results scenario 9B – Misspecified sparse partially distributed

α	γ	Frequency of minimum			Mean (SD) over 100 simulation iterations					Mean (SD) over 100 test obs.	
		Mean CVE	MSE($\hat{\beta}$)	MSE(\hat{y}_{test})	Mean CVE	SD CVE	MSE($\hat{\beta}$)	MSE(\hat{y}_{test})	Optimal λ	Bias ² ($\hat{y}_{\text{i test}}^*$)	Var($\hat{y}_{\text{i test}}^*$)
0.0	0.0	4	5	5	57.63 (18.10)	38.07 (18.88)	0.094 (0.079)	40.23 (32.95)	17.78 (100.05)	19.86 (37.68)	15.08 (11.39)
0.0	0.2	0	0	0	54.48 (14.43)	34.60 (15.68)	0.098 (0.068)	41.60 (26.84)	5.18 (11.44)	23.39 (45.40)	13.39 (9.13)
0.2	0.2	0	0	0	50.72 (14.36)	33.37 (15.50)	0.094 (0.072)	39.75 (28.07)	4.99 (9.34)	21.96 (42.93)	13.49 (10.12)
0.5	0.2	1	1	1	45.31 (14.56)	31.60 (15.50)	0.078 (0.069)	33.59 (27.01)	5.59 (9.42)	17.59 (34.53)	12.07 (10.10)
0.8	0.2	1	5	2	40.78 (15.06)	29.43 (15.19)	0.066 (0.064)	29.58 (26.39)	6.11 (11.53)	14.19 (28.09)	10.92 (9.65)
1.0	0.2	59	43	42	38.11 (15.14)	28.23 (15.38)	0.059 (0.062)	26.82 (25.76)	7.38 (18.76)	12.39 (24.60)	10.11 (9.29)
0.0	0.5	0	0	0	73.91 (15.43)	38.16 (17.32)	0.162 (0.056)	67.10 (23.91)	12.96 (99.87)	44.36 (86.98)	14.78 (5.67)
0.2	0.5	0	0	0	68.12 (14.82)	36.94 (16.79)	0.145 (0.063)	59.69 (24.95)	3.85 (9.75)	38.63 (76.24)	14.62 (6.70)
0.5	0.5	1	0	0	58.08 (14.67)	35.49 (15.60)	0.114 (0.073)	48.08 (30.29)	4.32 (10.00)	27.95 (54.86)	14.01 (9.34)
0.8	0.5	2	2	2	46.48 (15.79)	32.89 (15.22)	0.080 (0.070)	35.77 (29.84)	6.58 (14.87)	17.91 (35.22)	11.87 (9.44)
1.0	0.5	31	44	47	40.06 (16.45)	29.80 (14.84)	0.058 (0.064)	26.31 (25.13)	7.95 (12.39)	12.09 (23.73)	10.01 (9.11)
0.0	0.8	0	0	0	93.14 (19.77)	42.68 (18.60)	0.240 (0.036)	96.69 (23.70)	21.56 (103.07)	71.04 (136.17)	15.02 (3.10)
0.2	0.8	0	0	0	89.82 (18.83)	42.20 (18.52)	0.223 (0.041)	90.85 (23.12)	11.24 (27.28)	64.88 (125.21)	15.28 (3.70)
0.5	0.8	0	0	0	83.54 (17.86)	40.94 (18.32)	0.199 (0.052)	81.61 (24.11)	12.38 (32.14)	55.69 (107.58)	16.16 (5.00)
0.8	0.8	1	0	0	76.45 (18.25)	39.84 (18.52)	0.154 (0.072)	63.92 (31.37)	15.04 (30.60)	39.99 (78.20)	16.66 (8.22)
1.0	0.8	0	0	1	76.64 (20.42)	42.13 (20.47)	0.138 (0.098)	58.10 (41.60)	41.36 (115.35)	32.13 (62.27)	19.10 (14.95)
0.0	1.0	0	0	0	104.42 (22.92)	46.08 (19.97)	0.281 (0.034)	111.59 (25.27)	108.80 (282.70)	84.11 (158.93)	16.77 (3.90)
0.2	1.0	0	0	0	102.75 (22.40)	46.01 (19.58)	0.274 (0.035)	109.20 (25.26)	37.29 (71.43)	82.28 (155.68)	16.01 (3.63)
0.5	1.0	0	0	0	99.39 (21.04)	45.59 (19.68)	0.260 (0.037)	104.89 (25.43)	26.53 (34.41)	76.44 (144.45)	17.27 (3.90)
0.8	1.0	0	0	0	97.82 (21.17)	44.51 (18.96)	0.253 (0.047)	102.92 (27.24)	42.83 (51.51)	71.98 (135.72)	19.16 (5.69)
1.0	1.0	0	0	0	103.87 (23.26)	47.21 (18.98)	0.286 (0.050)	116.19 (30.34)	183.24 (302.75)	80.68 (152.09)	23.43 (6.85)

Notes:

- α controls balance between group lasso ($\alpha = 0$) and ℓ_1 lasso ($\alpha = 1$) penalty terms.
- γ controls balance between fusion ($\gamma = 0$) and sparsity ($\gamma = 1$) penalty terms.
- ‘Frequency of minimum’ columns give frequency of instances out of 100 of simulations yielding the minimum indicated error measure for each (α, γ) pair.
- Mean CVE (cross-validated error) is the minimum mean CVE over 5 cross-validation folds, which was used to select the optimal λ .
- MSE($\hat{\beta}$) is the mean squared error of the estimated coefficients for the model fitted to the entire training sample of $n = 50$.
- MSE(\hat{y}_{test}) is the mean squared error of the estimated response for a test sample of $n = 50$.
- Bias²($\hat{y}_{\text{i test}}^*$) and Var($\hat{y}_{\text{i test}}^*$) were calculated as follows: For each (α, γ) pair, the response for each observation in a test sample of $n = 100$ was calculated using each set of estimated coefficients from the 100 simulation iterations. The mean squared bias and variance of the predicted responses were calculated for each observation individually, and then the mean and standard deviations of these were calculated over the 100 observations.

True Coefficients 9B:
Misspecified Sparse Group



APPENDIX B

ABIDE APPLICATION RESULTS

Table B1: ABIDE dataset descriptive summary

	Overall ($n = 219$)	Training Set ($n = 175$)	Test Set ($n = 44$)	
	Mean (SD)	Mean (SD)	Mean (SD)	p^*
Age at scan	17.4 (7.5)	17.4 (7.3)	17.4 (8.4)	0.97
Full-scale IQ	108.6 (14.3)	108.2 (13.8)	110 (16.5)	0.50
Mean framewise displacement	0.09 (0.06)	0.09 (0.07)	0.08 (0.05)	0.71
Social Responsiveness Scale score	56.3 (42.7)	56.6 (43.1)	55.1 (41.8)	0.84
	n (%)	n (%)	n (%)	p^*
Diagnosis group				0.50
Autism spectrum disorder	111 (50.7)	91 (52.0)	20 (45.5)	
Typically developing	108 (49.3)	84 (48.0)	24 (54.5)	
Site of acquisition				0.95
Leuven_1	22 (10.0)	17 (9.7)	5 (11.4)	
NYU	101 (46.1)	82 (46.9)	19 (43.2)	
USM	72 (32.9)	57 (32.6)	15 (34.1)	
Yale	24 (11.0)	19 (10.9)	5 (11.4)	
Eye status at scan				0.09
Open	204 (93.2)	166 (94.9)	38 (86.4)	
Closed	15 (6.8)	9 (5.1)	6 (13.6)	

* Welch's t-test for continuous variables, Fisher's exact test for categorical variables

Note: One training set participant was missing mean framewise displacement data. In order to fit the linear regression model to compute adjusted SRS scores, this missing value was imputed by a linear regression fit to the remaining $n = 174$ training participants, modeling mean framewise displacement as outcome with predictors age, IQ, social responsiveness scale score, site of acquisition, and eye status at scan.

Table B2: Linear regression model for adjusting Social Responsiveness Scale scores
Fit to training set ($n = 175$).

	Estimate	Std. Error	t	p
Intercept	41.63	32.50	1.28	0.20
Age at scan	1.26	0.56	2.25	0.03
Full-scale IQ	-0.18	0.24	-0.77	0.45
Mean framewise displacement	130.09	50.93	2.55	0.01
Site of acquisition				
Leuven_1	reference			
NYU	9.74	12.42	0.78	0.43
USM	-5.88	11.78	-0.50	0.62
Yale	-12.09	15.15	-0.80	0.43
Eye status at scan				
Open	reference			
Closed	-1.65	15.04	-0.11	0.91

Note: One training set participant was missing mean framewise displacement data. In order to fit the linear regression model to compute adjusted SRS scores, this missing value was imputed by a linear regression fit to the remaining $n = 174$ training participants, modeling mean framewise displacement as outcome with predictors age, IQ, social responsiveness scale score, site of acquisition, and eye status at scan.

APPENDIX C

NORMAL AGING STUDY RESULTS

Table C1: Univariate feature screening of GMD data for predicting age

Region	MNI			Cluster size	Peak r
	x	y	z		
Right cerebellum	-15	84.7	-38.8	201	-0.587
Right parahippocampal gyrus	-18.6	1.7	-28.9	102	-0.665
Left cerebellum	11.1	83.6	-39.8	72	-0.557
Right cerebellum	-26.3	68.2	-26.8	67	-0.546
Right cerebellum	-38.7	79.2	-30.4	60	-0.507
Right medial temporal lobe	-35.2	-14.6	-43.8	50	-0.640
Right cerebellum	-16.4	77.9	-33.0	37	-0.573
Right hippocampus	-36.1	21.7	-14.9	37	-0.641
Left parahippocampal gyrus	18.9	0.9	-31.0	36	-0.542
Left cerebellum	16.4	79.1	-33.3	28	-0.650
Cerebellar vermis	-0.5	65.0	-37.8	26	-0.558
Right putamen	-25.1	-7.0	-4.8	24	-0.533
Right cerebellum	-36.3	71.6	-52.2	23	-0.473
Left cerebellum	14.2	71.9	-24.6	23	-0.488
Left inferior frontal gyrus	16.4	-6.9	-17.4	21	-0.569
Left cerebellum	27.4	66.2	-28.4	20	-0.485
Right cerebellum	-32.7	66.2	-42.0	18	-0.570
Right insula	-37.9	12.4	-3.9	16	-0.556
Right postcentral gyrus	-49.3	25.9	45.6	16	-0.555
Right cerebellum	-7.2	72.0	-20.5	15	-0.584
Left cerebellum	50.4	61.0	-34.6	13	-0.461
Right superior temporal gyrus / temporal pole	-36.9	-14.0	-26.6	13	-0.538
Left anterior cingulate cortex	2.0	-30.7	16.3	13	-0.547
Right fusiform gyrus	-40.8	12.5	-32.5	12	-0.528
Right middle temporal gyrus	-52.7	25.8	-6.0	12	-0.588
Right cerebellum	-49.1	69.6	-32.9	11	-0.502
Right parahippocampal gyrus / amygdala	-26.5	3.8	-23.2	11	-0.513
Right medial frontal / rectal gyrus	0.4	-26.7	-17.5	11	-0.486
Right inferior frontal gyrus	-49.5	-35.1	-14.3	11	-0.561
Left cerebellum	13.2	86.2	-28.8	10	-0.518

Threshold $|r| \geq 0.4145$, 1009 voxels, minimum cluster size 10.

MNI coordinates are for the center of mass of each cluster.

GMD: grey matter density

Table C2: Univariate feature screening of DMN/ASN data for predicting age

Region	MNI			Cluster size	Peak r
	x	y	z		
Left anterior cingulate cortex	1.9	-51.2	13.4	213	-0.455
Left medial frontal gyrus	2.5	-39.0	-13.3	176	-0.469
Left cingulate gyrus	0.8	17.8	33.4	121	-0.531
Right middle occipital / inferior temporal gyrus	-52.9	56.6	-6.8	98	0.423
Left superior frontal gyrus	11.4	-62.3	30.0	95	-0.528
Right superior temporal gyrus	-50.1	11.5	1.5	89	-0.437
Right cingulate gyrus	-20.1	13.0	44.4	79	-0.497
Right inferior frontal gyrus / insula (white matter)	-28.0	-18.5	22.0	65	-0.502
Right superior parietal lobule	-17.3	69.9	56.5	64	0.439

Threshold $|r| \geq 0.3363$, 1000 voxels, minimum cluster size 50.

MNI coordinates are for the center of mass of each cluster.

GMD: grey matter density

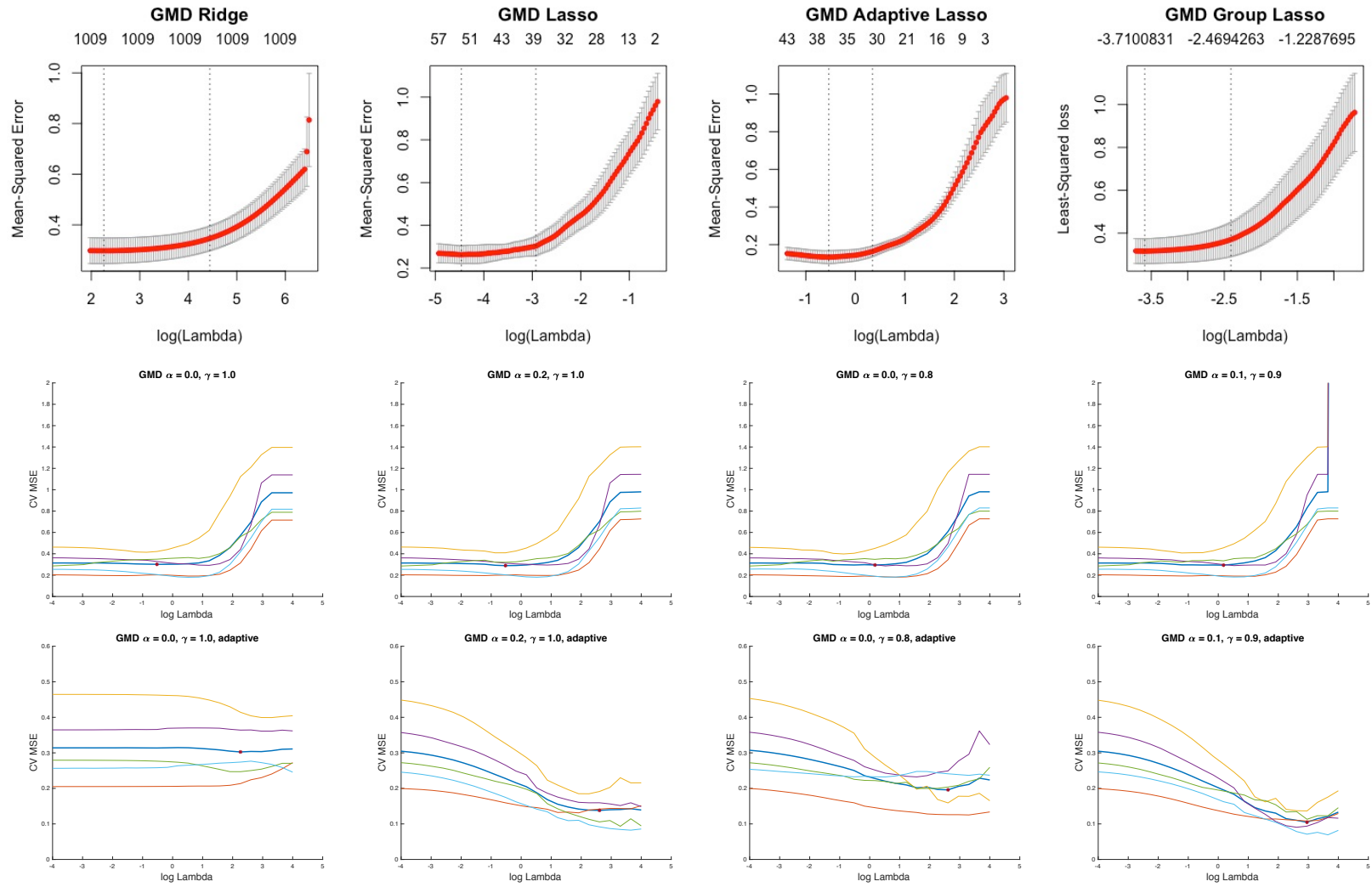


Figure C1: Normal Aging unimodal GMD cross-validation error curves. For FSGL penalties, red asterisks mark minimum cross-validation mean squared error.

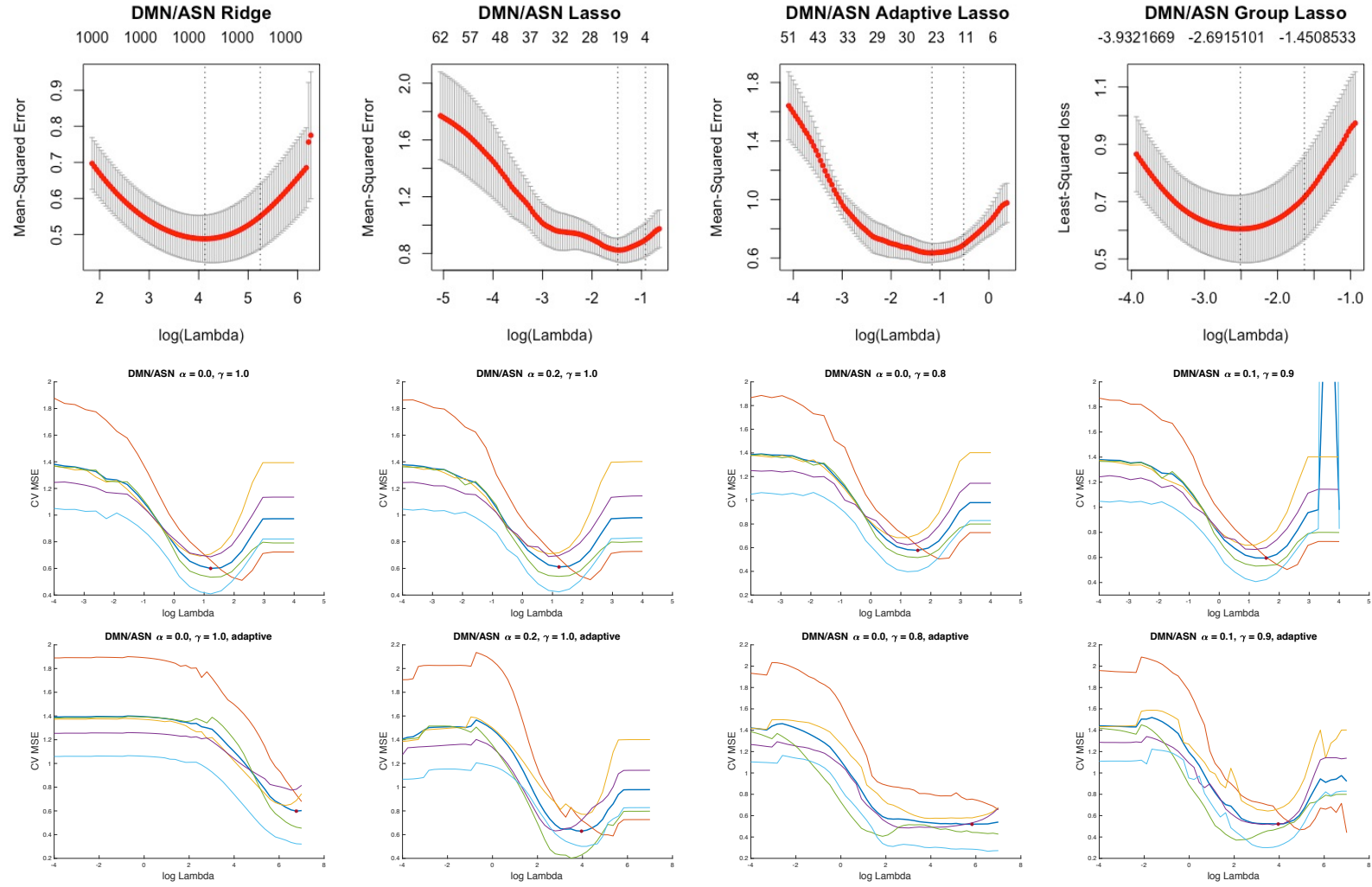


Figure C2: Normal Aging unimodal DMN/ASN cross-validation error curves. For FSGL penalties, red asterisks mark minimum cross-validation mean squared error.

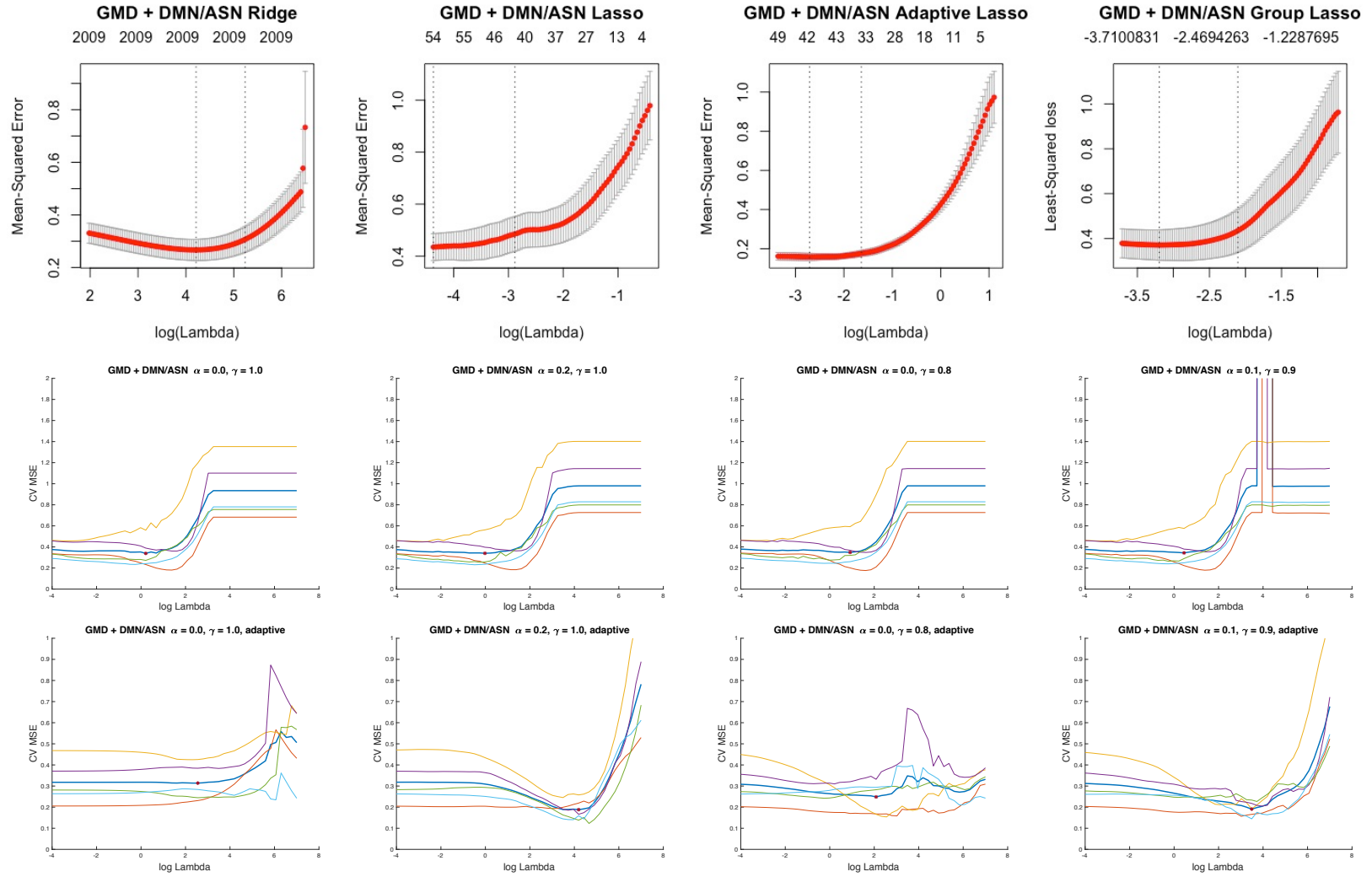


Figure C3: Normal Aging multimodal GMD and DMN/ASN cross-validation error curves. For FSGL penalties, red asterisks mark minimum cross-validation mean squared error.

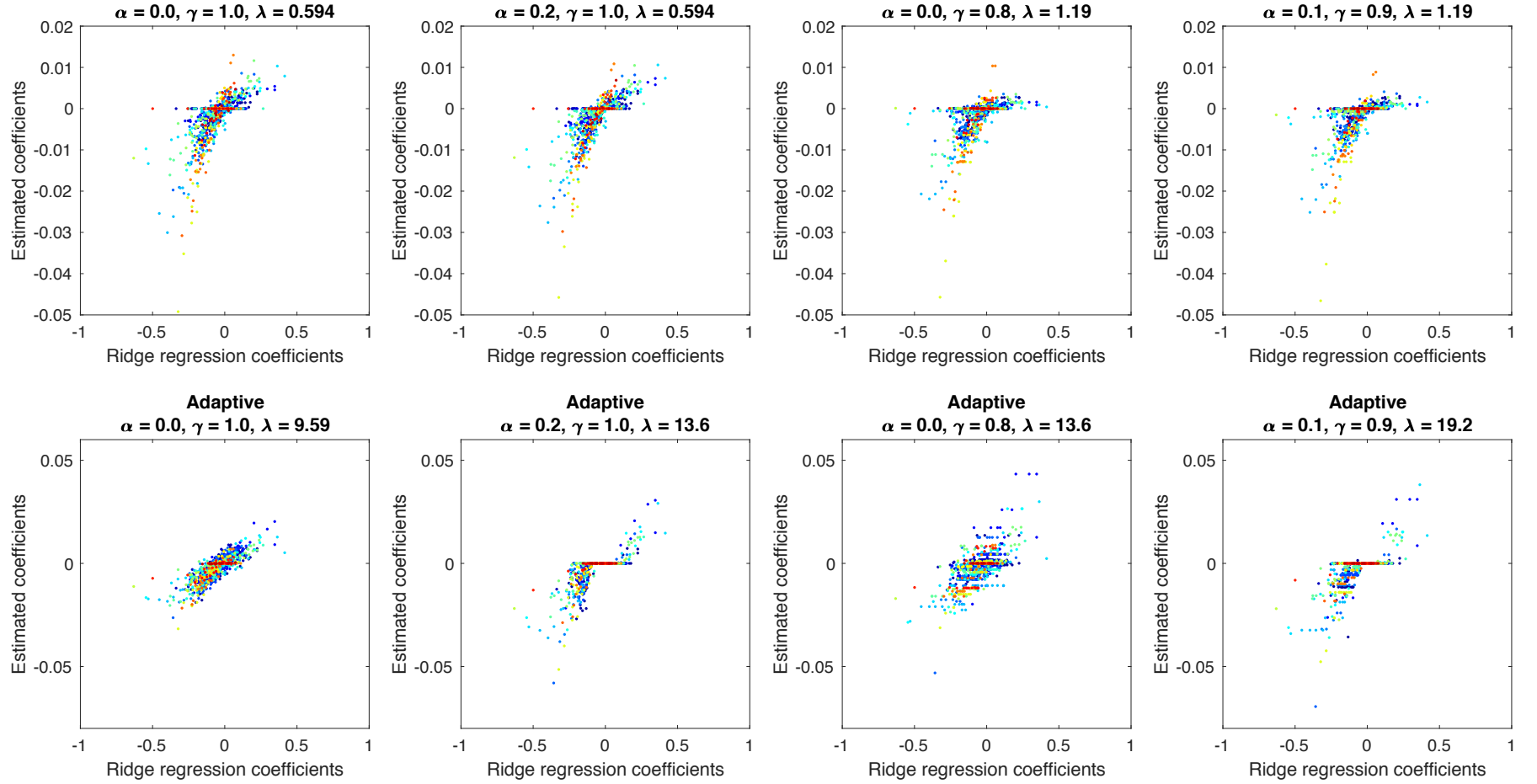


Figure C4: Normal Aging unimodal GMD estimated coefficients. For various fused sparse group lasso estimators, plotted against the ridge regression estimated coefficients. Plotting points are colored by voxel group membership.

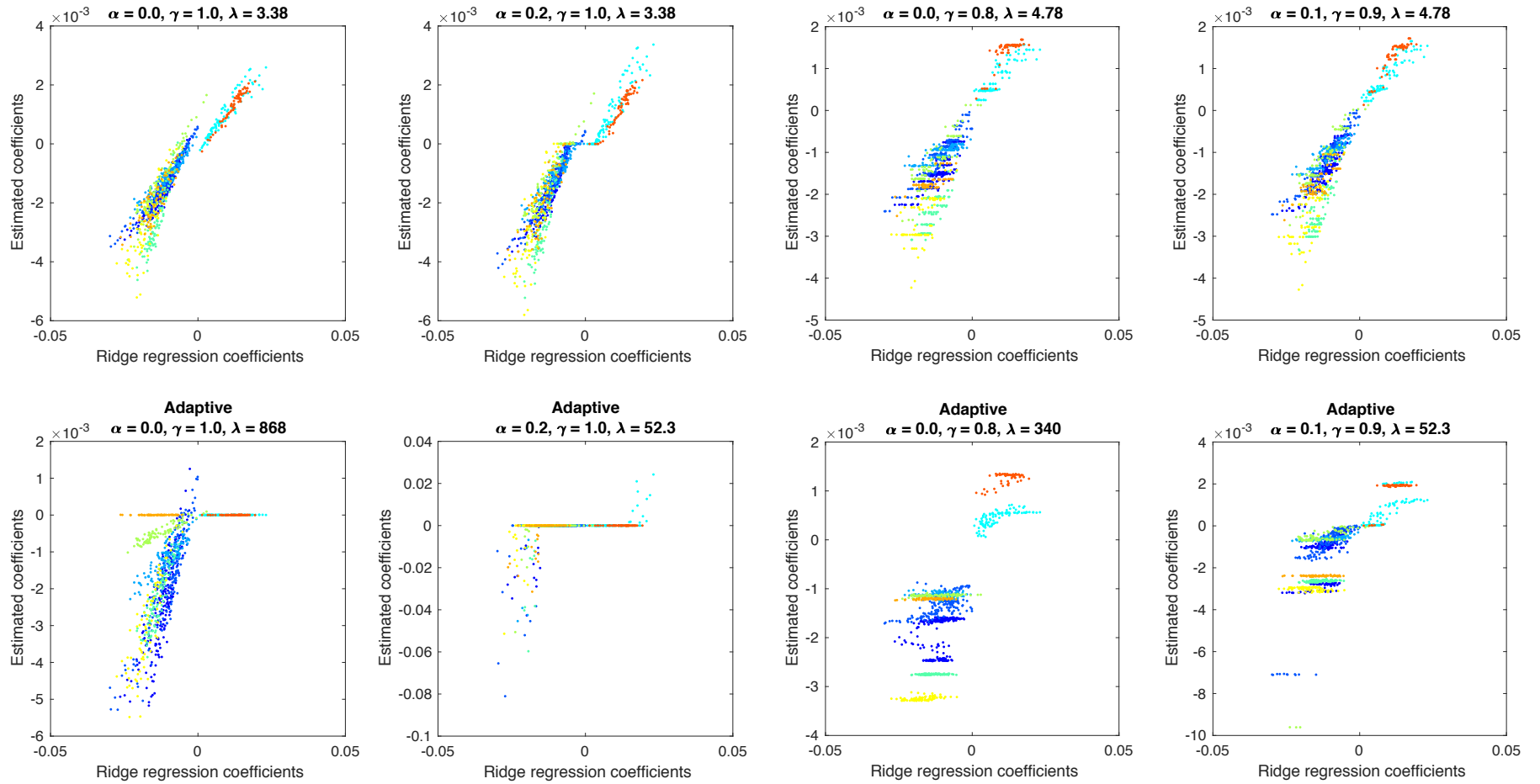


Figure C5: Normal Aging unimodal DMN/ASN estimated coefficients. For various fused sparse group lasso estimators, plotted against the ridge regression estimated coefficients. Plotting points are colored by voxel group membership.

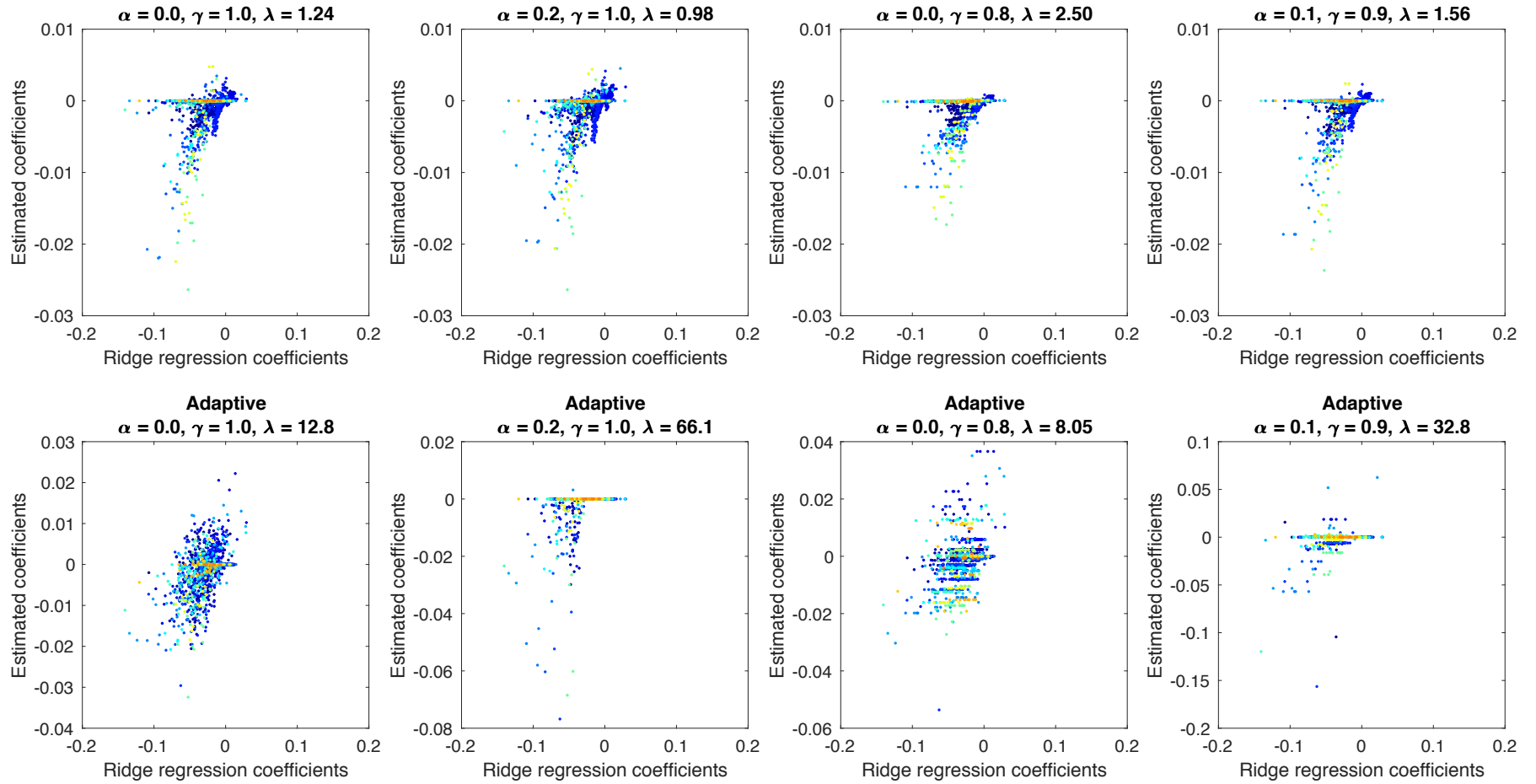


Figure C6: Normal Aging multimodal GMD and DMN/ASN estimated coefficients. For various fused sparse group lasso estimators, plotted against the ridge regression estimated coefficients. Plotting points are colored by voxel group membership.

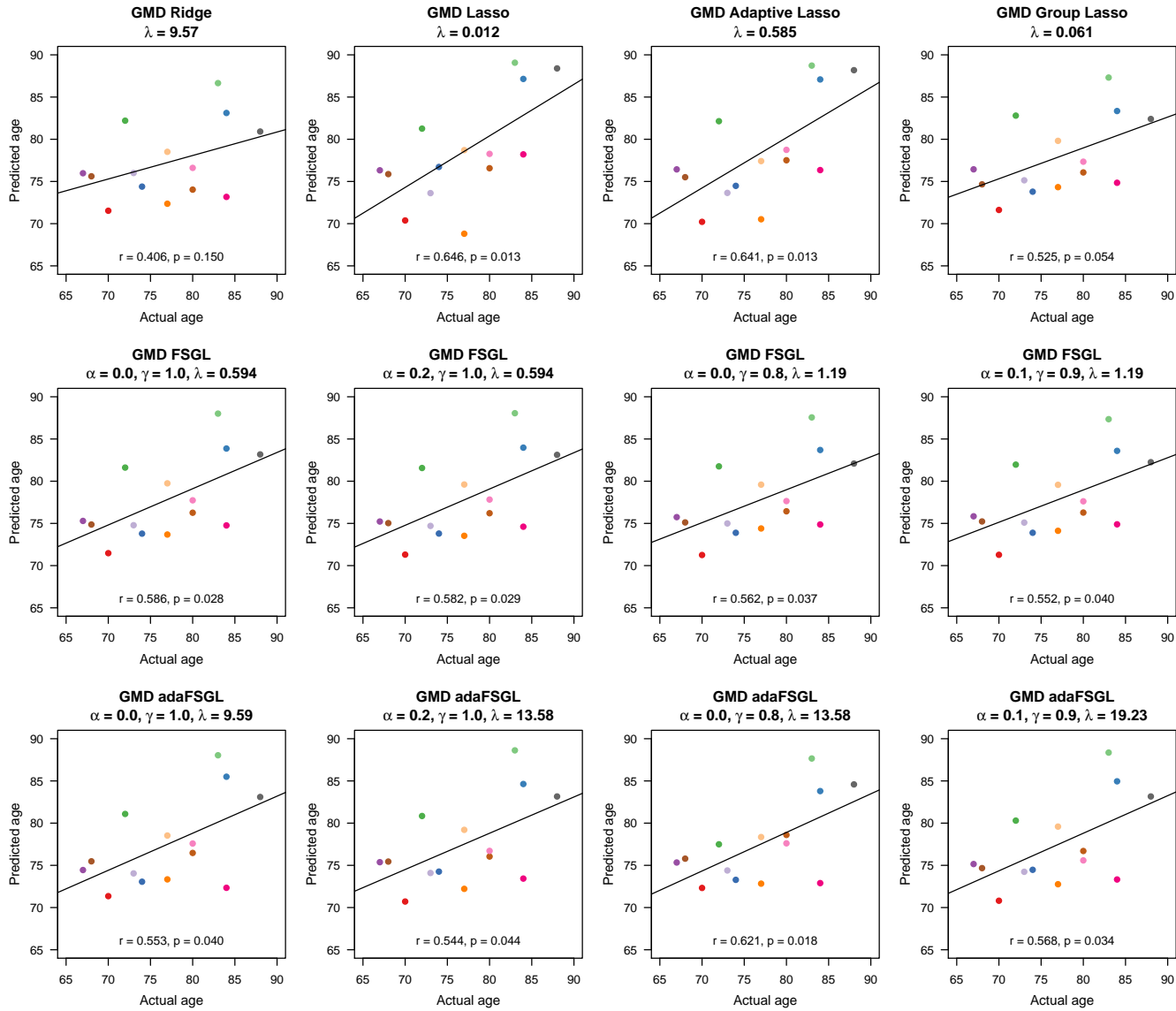


Figure C7: Normal Aging unimodal GMD predicted versus actual age

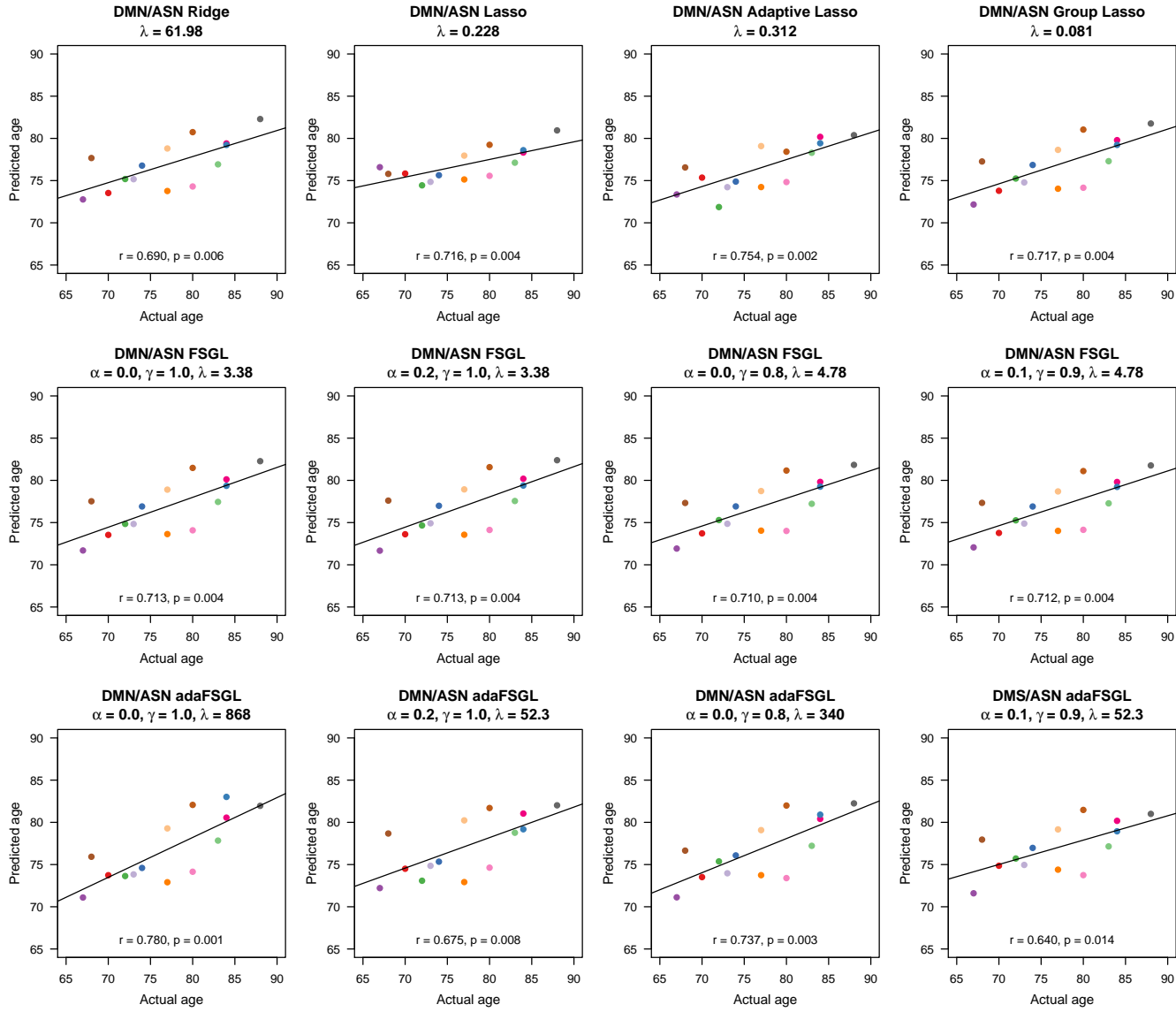


Figure C8: Normal Aging unimodal DMS/ASN predicted versus actual age

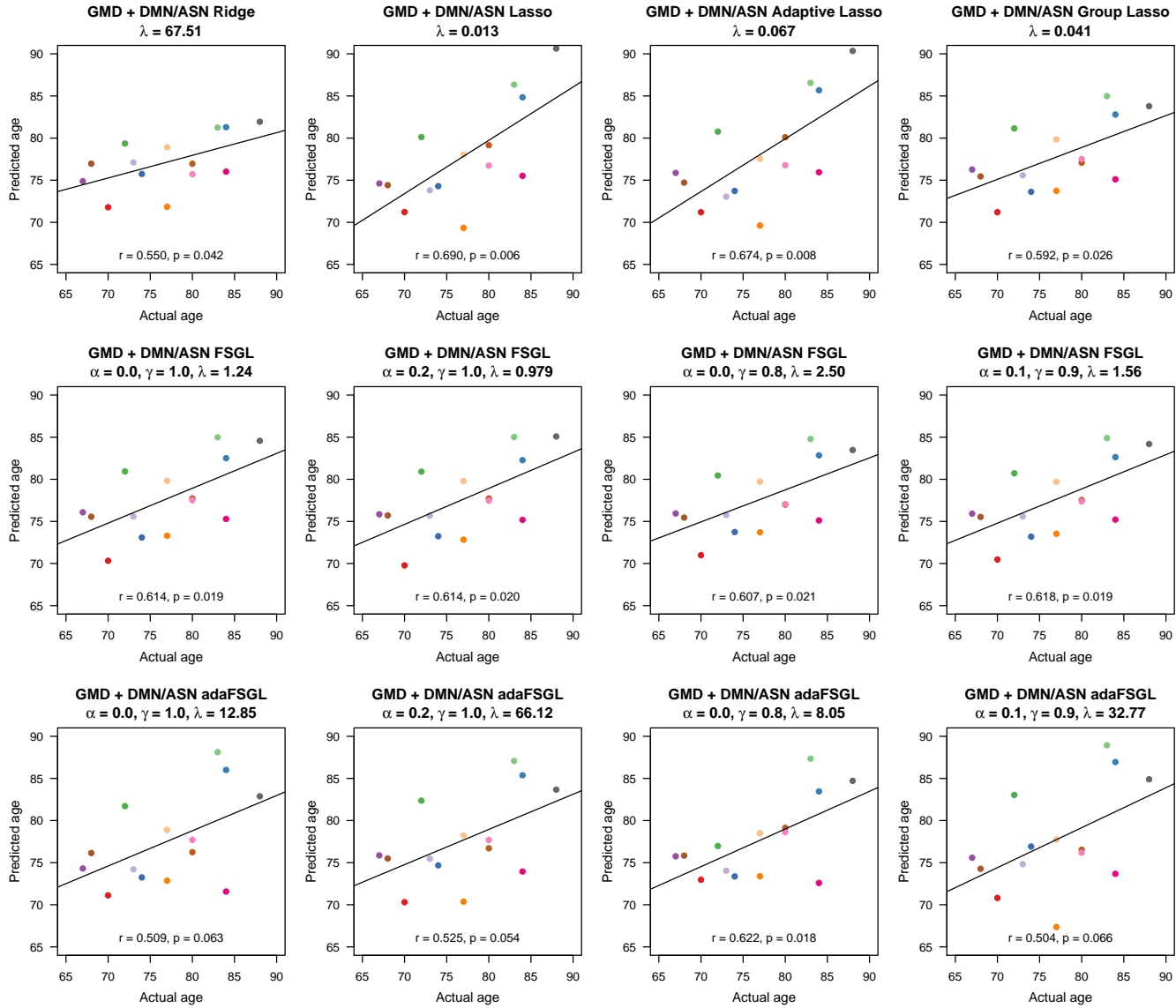


Figure C9: Normal Aging multimodal GMD and DMS/ASN predicted versus actual age

APPENDIX D

R CODE

Note: The following R code as well as additional code used in the simulation study described in Section 2.3 and the ABIDE application described in Section 2.4 is available online at <https://github.com/jcbeer/fsgl>.

D.1 R FUNCTION: `softthresh.R`

```
1 # Soft-threshold a scalar or vector.
2 ### INPUTS
3 # a: a scalar or vector.
4 # kappa: soft-thresholding parameter.
5 ### EXAMPLE
6 # softthresh(c(3, 4), 1)
7
8 softthresh <- function(a, kappa){
9   if(sum(a == 0) == length(a)){
10     return(a)
11   } else {
12     return(max(0, (1 - kappa/sqrt(sum(a^2))))*a)
13   }
14 }
```

D.2 R FUNCTION: makeKmatrix2d.R

```
1 # Builds the K matrix for fused sparse group lasso fsgl.fit.R function where graph structure is 2D fused. Requires Matrix package.
2 ### INPUTS
3 # dim1: number of rows of image.
4 # dim2: number of columns of image.
5 # groups: vector encoding group membership, with groups indexed as 1, 2, etc. Note: columns of image are concatenated from left to right, group
6 # membership coding should correspond.
7 ### EXAMPLE
8 # makeKmatrix2d(dim1=3, dim2=3, groups=c(1, 2, 3, 2, 2, 1, 3, 3, 1))
9 makeKmatrix2d <- function(dim1, dim2, groups){
10   ### make J matrix for lasso
11   p <- dim1*dim2
12   J <- diag(p)
13   ### make D matrix for 2D fused lasso
14   # matrix block to fuse one column
15   D.fuse.column.vector <- rep(0, dim1*(dim1-1))
16   D.fuse.column.vector[(0:(dim1-2)*(dim1) + 1:(dim1-1))] <- -1
17   D.fuse.column.vector[(0:(dim1-2)*(dim1) + 2:(dim1))] <- 1
18   D.fuse.column <- matrix(D.fuse.column.vector, nrow=(dim1-1), ncol=dim1, byrow=TRUE)
19   # make this into a block diagonal matrix
20   D.fuse.column.list <- paste0('list(', paste(rep('D.fuse.column', dim2), collapse=', '), ')')
21   big.D.column <- Matrix::bdiag(eval(parse(text=D.fuse.column.list)))
22   # matrix block to fuse rows
23   big.D.row <- matrix(0, nrow=(dim2-1)*dim1, ncol=p)
24   big.D.row.neg.ones <- 0:(dim2-2)*dim1 + rep(1:dim1, each=(dim2-1))
25   big.D.row.ones <- 1:(dim2-1)*dim1 + rep(1:dim1, each=(dim2-1))
26   for(i in 1:dim(big.D.row)[1]){
27     big.D.row[i, big.D.row.neg.ones[i]] <- -1
28     big.D.row[i, big.D.row.ones[i]] <- 1
29   }
30   # now bind these together
31   D <- rbind(big.D.column, big.D.row)
32   D <- as.matrix(D)
33   ### make G matrix for groups
34   G <- diag(groups==1)
35   for(i in 2:max(groups)){
36     G <- rbind(G, diag(groups==i))
37   }
38   # remove zero rows
39   G <- G[rowSums(G)==1,]
40   # put together into K matrix
41   K <- rbind(J, D, G)
42   return(list(K=K, nj=nrow(J), nd=nrow(D), ngroups=max(groups), groupssize=as.vector(table(groups))))
43 }
```

D.3 R FUNCTION: makeKmatrix3d.R

```

1 # Builds the K matrix for fused sparse group lasso fsgl.fit.R function where graph structure is 3D fused. Requires Matrix package.
2 ### INPUTS
3 # dim1: dimension along x axis.
4 # dim2: dimension along y axis.
5 # dim3: dimension along z axis.
6 # mask: binary vector indicating masked areas (1s) zeros (0s).
7 # groups: vector encoding group membership, with groups indexed as 1, 2, etc. If there is a mask, put zeros or other placeholder at location of
   masked voxels. Note: columns of each slice of image are concatenated from left to right, group membership coding should correspond.
8 ### EXAMPLE
9 Kmatrix <- makeKmatrix3d(dim1=2, dim2=3, dim3=4, mask=c(rep(0, 4), rep(1, 16), rep(0, 4)), groups=c(rep(0, 4), rep(1, 4), rep(2, 4), rep(3, 4),
   rep(4, 4), rep(0, 4)))
10
11 makeKmatrix3d <- function(dim1, dim2, dim3, mask=NULL, groups){
12   ### make J matrix for lasso penalty
13   if(!is.null(mask)){
14     J <- diag(sum(mask))
15   } else {
16     J <- diag(dim1*dim2*dim3)
17   }
18   ### make D matrix for 3D fused lasso
19   # matrix block to fuse one column
20   D.fuse.column.vector <- rep(0, dim1*(dim1-1))
21   D.fuse.column.vector[(0:(dim1-2)*(dim1) + 1:(dim1-1))] <- -1
22   D.fuse.column.vector[(0:(dim1-2)*(dim1) + 2:(dim1))] <- 1
23   D.fuse.column <- matrix(D.fuse.column.vector, nrow=(dim1-1), ncol=dim1, byrow=TRUE)
24   # make this into a block diagonal matrix
25   D.fuse.column.list <- paste0('list(', paste(rep('D.fuse.column', dim2), collapse=', '), ')')
26   big.D.column <- Matrix::bdiag(eval(parse(text=D.fuse.column.list)))
27   # remove some stuff
28   rm(list=c('D.fuse.column.vector', 'D.fuse.column', 'D.fuse.column.list'))
29   # matrix block to fuse rows
30   big.D.row <- matrix(0, nrow=(dim2-1)*dim1, ncol=dim1*dim2)
31   big.D.row.neg.ones <- 0:(dim2-2)*dim1 + rep(1:dim1, each=(dim2-1))
32   big.D.row.ones <- 1:(dim2-1)*dim1 + rep(1:dim1, each=(dim2-1))
33   for(i in 1:dim(big.D.row)[1]){
34     big.D.row[i, big.D.row.neg.ones[i]] <- -1
35     big.D.row[i, big.D.row.ones[i]] <- 1
36   }
37   # remove some stuff
38   rm(list=c('big.D.row.neg.ones', 'big.D.row.ones'))
39   # now bind these together and make a block diagonal
40   D.row.col <- rbind(big.D.column, big.D.row)
41   rm(list=c('big.D.column', 'big.D.row'))
42   D.row.col.list <- paste0('list(', paste(rep('D.row.col', dim3), collapse=', '), ')')
43   big.D.row.col <- Matrix::bdiag(eval(parse(text=D.row.col.list)))
44   # remove some stuff
45   rm(list=c('D.row.col', 'D.row.col.list'))
46   # matrix block to fuse slices
47   D.slice <- diag(dim1*dim2*dim3)[1:(dim1*dim2*(dim3-1)),]
48   neg.ones <- (1 + dim1*dim2):(dim1*dim2*dim3)
49   D.slice[cbind(1:(dim1*dim2*(dim3-1)), neg.ones)] <- -1
50   # add this to existing D matrix
51   D <- rbind(big.D.row.col, D.slice)
52   # remove some stuff
53   rm(list=c('big.D.row.col', 'D.slice', 'neg.ones'))
54   D <- as.matrix(D)
55   # remove non-masked
56   if(!is.null(mask)){
57     keep.rows <- which(rowSums(abs(D[,mask==0])) == 0)
58     D <- D[keep.rows, mask==1]
59   }
60   ### make G matrix for groups
61   if(!is.null(mask)){
62     groups <- groups[mask==1]
63   }
64   G <- diag(groups==1)*1
65   for(i in 2:max(groups)){
66     G <- rbind(G, diag(groups==i)*1)
67   }
68   # remove zero rows
69   G <- G[rowSums(G)==1,]
70   # put together into K matrix
71   K <- rbind(J, D, G)
72   return(list(K=K, nj=nrow(J), nd=nrow(D), ngroups=max(groups), group sizes=as.vector(table(groups))))
73 }

```

D.4 R FUNCTION: fsgl.fit.R

```

1 # Fits a regression model for one set of tuning parameter values by minimizing a penalized least squares loss function.
2 # INPUTS
3 # X: a n*p matrix of predictor variables with observations in rows.
4 # Y: a n*1 vector of the response variable.
5 # K: a (nj + nd + ng)*p matrix encoding the lasso penalty (first nj rows), the graph structure for the fused penalty (the next nd rows), and the
   group structure for the group penalty (last ng rows). Can be made with function makeKmatrix.
6 # nj: number of rows of K that encode the lasso penalty. If lasso penalty is applied to all coefficients then this will equal p.
7 # nd: number of rows of K that encode the graph structure for the fused penalty.
8 # ngroups: number of groups for the group penalty.
9 # groupsizes: a vector of length ngroups that gives the size of each group in the order they appear in the K matrix. Sum should equal ng.
10 # alpha: tuning parameter that controls the degree of group (alpha = 0) vs L1 (alpha=1) sparsity. 0 <= alpha <= 1
11 # gamma: tuning parameter that controls the degree of sparsity (gamma=1) vs fusion (gamma=0) penalty. 0 <= gamma <= 1
12 # lambda: tuning parameter that controls the overall degree of regularization.
13 # beta0: starting values for beta. Defaults to zero vector.
14 # theta0: starting values for theta. Defaults to zero vector.
15 # mu0: starting values for mu. Defaults to zero vector.
16 # beta_update_factor: by default function will set this equal to solve(t(Xcentered) %*% Xcentered + rho * t(K) %*% K). It may take a long time to
   compute and does not depend on tuning parameters, so it may be provided as an argument if fitting the model at many sets of tuning parameters
17 # beta_update_term: by default function will set this equal to t(Xcentered) %*% Ycentered. Does not depend on tuning parameters, so it may be
   provided as an argument if fitting the model at many sets of tuning parameters.
18 # rho: step size for ADMM algorithm. Defaults to 1.
19 # Niter: number of ADMM iterations. Defaults to 2000.
20 # epsilon_abs: absolute tolerance for ADMM convergence. Defaults to 10^-3.
21 # epsilon_rel: relative tolerance for ADMM convergence. Defaults to 10^-3.
22 # verbose: if TRUE will print number of ADMM iterations used. Defaults to FALSE.
23 ### EXAMPLE
24 # Kmatrix <- makeKmatrix2d(dim1=3, dim2=3, groups=c(1, 2, 3, 2, 2, 1, 3, 3, 1))
25 # x <- matrix(rnorm(54), nrow=6)
26 # y <- x %*% c(0, 0, 10, 0, 0, 0, 0, 10, 10, 0)
27 # fsgl.fit(X=x, Y=y, K=Kmatrix$K, nj=Kmatrix$nj, nd=Kmatrix$nd, ngroups=Kmatrix$ngroups, groupsizes=Kmatrix$groupsizes, alpha=0.1, gamma=0.8,
   lambda=5)
28
29 fsgl.fit <-
30   function(X,
31     Y,
32     K,
33     nj,
34     nd,
35     ngroups,
36     groupsizes,
37     alpha,
38     gamma,
39     lambda,
40     beta0 = NULL,
41     theta0 = NULL,
42     mu0 = NULL,
43     beta_update_factor = NULL,
44     beta_update_term1 = NULL,
45     rho = 1,
46     Niter = 2000,
47     epsilon_abs = 10 ^ -3,
48     epsilon_rel = 10 ^ -3,
49     verbose = FALSE) {
50   # center the columns of X and y
51   # so we won't have to estimate an intercept
52   Xcentered <- scale(X, center = TRUE, scale = FALSE)
53   Ycentered <- scale(Y, center = TRUE, scale = FALSE)
54   # make a list of 'group' indices for theta update step
55   # where 'group' includes all lasso, fused lasso, and group lasso terms
56   group_indices <- list()
57   ni <- c(rep(1, nj), rep(1, nd), groupsizes) # vector of group sizes
58   cumsumni <- cumsum(ni)
59   N <- nj + nd + ngroups
60   for (i in 1:N){
61     group_indices[[i]] <- (max(0, cumsumni[i-1]) + 1):cumsumni[i]
62   }
63   # if not supplied as arguments then
64   # calculate beta update terms
65   # which don't change over loop iterations
66   if (is.null(beta_update_factor))
67     beta_update_factor <-
68     solve(t(Xcentered) %*% Xcentered + rho * t(K) %*% K)
69   if (is.null(beta_update_term1))
70     beta_update_term1 <- t(Xcentered) %*% Ycentered

```

```

71 | # set initial values to zero if not specified
72 | if (is.null(beta0))
73 |   beta0 <- rep(0, ncol(K))
74 | if (is.null(theta0))
75 |   theta0 <- rep(0, nrow(K))
76 | if (is.null(mu0))
77 |   mu0 <- rep(0, nrow(K))
78 | # initialize parameters
79 | current_beta <- beta0
80 | current_theta <- theta0
81 | current_mu <- mu0
82 | # define lambda vector
83 | Lambda <- c(rep(alpha * gamma * lambda, nj),
84 |             rep((1 - gamma) * lambda, nd),
85 |             rep((1 - alpha) * gamma * lambda, ngroups))
86 | # LOOP over ADMM iterations
87 | for (t in 1:Niter) {
88 |   # UPDATE PARAMETERS
89 |   # update beta
90 |   beta_update_term2 <-
91 |     t(K) %*% matrix((current_mu - rho * current_theta), ncol = 1)
92 |   current_beta <-
93 |     beta_update_factor %*% (beta_update_term1 - beta_update_term2)
94 |   # update theta
95 |   eta <- K %*% current_beta + current_mu / rho
96 |   previous_theta <- current_theta
97 |   for (i in 1:N) {
98 |     current_theta[group_indices[[i]]] <-
99 |       softthresh(eta[group_indices[[i]]], kappa = (Lambda[i] * sqrt(length(eta[group_indices[[i]]]))) /
100 |        rho)
101 |   }
102 |   # update mu
103 |   current_mu <- current_mu + rho * (K %*% current_beta - current_theta)
104 |   # CHECK STOPPING CONDITIONS
105 |   # calculate dual residual
106 |   epsilon_dual <-
107 |     sqrt(length(current_theta)) * epsilon_abs + epsilon_rel * sqrt(sum((t(K) %*% current_mu) ^ 2))
108 |   current_s <-
109 |     sqrt(sum((rho * t(K) %*% (previous_theta - current_theta)) ^ 2))
110 |   # calculate primal residual
111 |   epsilon_primal <-
112 |     sqrt(length(current_beta)) * epsilon_abs + epsilon_rel * max(sqrt(sum((K %*% current_beta) ^ 2)), sqrt(sum(current_theta ^ 2)))
113 |   current_r <- sqrt(sum((K %*% current_beta - current_theta) ^ 2))
114 |   # break loop if conditions are met
115 |   if (current_s < epsilon_dual &
116 |       current_r < epsilon_primal)
117 |     break
118 | } # end loop over ADMM iterations
119 | if (verbose == TRUE) print(t)
120 | # save outputs
121 | pred.y <- Ycentered %*% current_beta
122 | mse.y <- mean((Ycentered - pred.y) ^ 2)
123 | output <-
124 |   list(
125 |     pred.y = pred.y,
126 |     mse.y = mse.y,
127 |     beta = current_beta,
128 |     theta = current_theta,
129 |     mu = current_mu,
130 |     niter = t
131 |   )
132 | return(output)
133 | }

```


D.5 R FUNCTION: fsgl.cv.R

```

1 # Does k-fold validation for fsgl.fit.R.
2 ### INPUTS
3 # X: a n*p matrix of predictor variables with observations in rows.
4 # Y: a n*1 vector of the response variable.
5 # K: a (nj + nd + ng)*p matrix encoding the lasso penalty (first nj rows), the graph structure for the fused penalty (the next nd rows), and the
   group structure for the group penalty (last ng rows). Can be made with function \code{makeKmatrix}.
6 # nj: number of rows of K that encode the lasso penalty. If lasso penalty is applied to all coefficients then this will equal p.
7 # nd: number of rows of K that encode the graph structure for the fused penalty.
8 # ngroups: number of groups for the group penalty.
9 # groupsizes: a vector of length ngroups that gives the size of each group in the order they appear in the K matrix. Sum should equal ng.
10 # alphagamma: a two-column matrix with rows containing alpha, gamma pairs at which cross-validation will be done for a range of lambda values.
   alpha is a tuning parameter that controls the degree of group (alpha = 0) vs L1 (alpha=1) sparsity. gamma is a tuning parameter that controls
   the degree of sparsity (gamma=1) vs fusion (gamma=0) penalty. 0 <= alpha, gamma <= 1
11 # lambda: vector of decreasing lambda values at which k-fold cross-validation will be done. tuning parameter that controls the overall degree of
   regularization.
12 # k: number of folds for cross-validation.
13 # folds: a vector encoding the fold assignments. By default folds will be randomly assigned.
14 # beta0: starting values for beta. Defaults to zero vector.
15 # theta0: starting values for theta. Defaults to zero vector.
16 # mu0: starting values for mu. Defaults to zero vector.
17 # rho: step size for ADMM algorithm. Defaults to 1.
18 # Niter: number of ADMM iterations used for each fsgl fit. Defaults to 2000.
19 # epsilon_abs: absolute tolerance for ADMM convergence. Defaults to 10^-3.
20 # epsilon_rel: relative tolerance for ADMM convergence. Defaults to 10^-3.
21 # verbose: if TRUE will print progress through alpha, gamma combinations for each fold. Defaults to FALSE.
22 ### EXAMPLE
23 # Kmatrix <- makeKmatrix2d(dim1=3, dim2=3, groups=c(1, 2, 3, 2, 2, 1, 3, 3, 1))
24 # x <- matrix(rnorm(54), nrow=6)
25 # y <- x %*% c(0, 0, 10, 0, 0, 0, 0, 10, 10, 0)
26 # # define lambda
27 # Lambda <- 10 ^ seq(3, -3, length = 50)
28 # # define alpha and gamma
29 # alphas <- c(rep(0.5), rep(0.2, 4), rep(0.5, 4), rep(0.8, 4), rep(1, 4))
30 # gammas <- c(0, rep(c(0.2, 0.5, 0.8, 1), 5))
31 # AlphaGamma <- cbind(alphas, gammas)
32 # cv <- fsgl.cv(X=x, Y=y, K=Kmatrix$K, nj=Kmatrix$nj, nd=Kmatrix$nd, ngroups=Kmatrix$ngroups, groupsizes=Kmatrix$groupsizes, alphagamma=AlphaGamma,
   lambda=Lambda, k=3, verbose=TRUE)
33
34 fsgl.cv <-
35   function(X,
36     Y,
37     K,
38     nj,
39     nd,
40     ngroups,
41     groupsizes,
42     alphagamma,
43     lambda,
44     k,
45     folds = NULL,
46     beta0 = NULL,
47     theta0 = NULL,
48     mu0 = NULL,
49     rho = 1,
50     Niter = 2000,
51     epsilon_abs = 10 ^ -3,
52     epsilon_rel = 10 ^ -3,
53     verbose = FALSE) {
54   if (is.null(folds)) folds <- sample(rep_len(1:k, nrow(X)))
55   # create array to save results
56   results <- array(dim=c(nrow(alphagamma), length(lambda), k))
57   # LOOP over folds
58   for (fold in 1:k){
59     if (verbose == TRUE) print(paste0('Starting Fold ', fold))
60     # select in and out of CV fold samples and center the data
61     curX <- scale(X[folds != fold,], center=TRUE, scale=FALSE)
62     curY <- scale(Y[folds != fold], center=TRUE, scale=FALSE)
63     curXout <- scale(X[folds == fold,], center=TRUE, scale=FALSE)
64     curYout <- scale(Y[folds == fold], center=TRUE, scale=FALSE)
65     # calculate beta update terms for CV sample
66     beta_update_factor_cv <- solve(t(curX) %*% curX + rho * t(K) %*% K)
67     beta_update_term1_cv <- t(curX) %*% curY
68     # LOOP over alpha, gamma pairs
69     for (i in 1:nrow(alphagamma)){
70       # set alpha and gamma

```

```

71 alpha <- alphagamma[i, 1]
72 gamma <- alphagamma[i, 2]
73 if (verbose == TRUE) print(paste0('alpha = ', alpha, ', gamma = ', gamma))
74 # initialize parameters
75 # start all parameters at zero when lambda is largest
76 if (is.null(beta0))
77   beta0 <- rep(0, ncol(K))
78 if (is.null(theta0))
79   theta0 <- rep(0, nrow(K))
80 if (is.null(mu0))
81   mu0 <- rep(0, nrow(K))
82 current_beta <- beta0
83 current_theta <- theta0
84 current_mu <- mu0
85 # LOOP over lambda values
86 for (j in 1:length(lambda)){
87   # fit estimator to the current CV sample
88   fit <-
89     fsgl.fit(
90       X=curX,
91       Y=curY,
92       K=K,
93       nj=nj,
94       nd=nd,
95       ngroups=ngroups,
96       groupsizes=groupsizes,
97       alpha=alpha,
98       gamma=gamma,
99       lambda=lambda[j],
100      beta0 = current_beta,
101      theta0 = current_theta,
102      mu0 = current_mu,
103      beta_update_factor = beta_update_factor_cv,
104      beta_update_term1 = beta_update_term1_cv,
105      rho = 1,
106      Niter = 2000,
107      epsilon_abs = 10 ^ -3,
108      epsilon_rel = 10 ^ -3,
109      verbose = TRUE
110    )
111   # calculate and save the out of sample MSE
112   pred.y.cv <- curYout %*% fit$beta
113   results[i, j, fold] <- mean((curYout - pred.y.cv) ^ 2)
114   # update the parameters for warm start at next lambda value
115   current_beta <- fit$beta
116   current_theta <- fit$theta
117   current_mu <- fit$mu
118 } # end loop over lambda values
119 } # end loop over alpha, gamma pairs
120 } # end loop over folds
121 # determine optimal lambda for each alpha, gamma pair
122 # calculate mean and sd cve across folds
123 mean.cve <- apply(results, c(1, 2), mean)
124 sd.cve <- apply(results, c(1, 2), sd)
125 # find which lambda is minimum for each alpha, gamma pair
126 min.cve.index <- apply(mean.cve, 1, which.min)
127 opt.lambda <- lambda[min.cve.index]
128 opt.mean.cve <- apply(mean.cve, 1, function(x) x[which.min(x)])
129 opt.sd.cve <- sd.cve[cbind(1:nrow(sd.cve), min.cve.index)]
130 optimal.lambdas <- cbind(alphagamma, opt.lambda, opt.mean.cve, opt.sd.cve)
131 return(list(optimal.lambdas=optimal.lambdas, mean.cve=mean.cve, sd.cve=sd.cve))
132 }

```

APPENDIX E

MATLAB CODE

Note: The following MATLAB code and additional code used for the ABIDE application described in Section 2.4 is available online at <https://github.com/jcbeer/fsgl>.

E.1 MATLAB FUNCTION: makeKmatrix.m

```
1 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2 % MAKEKMATRIX
3 % Function to create K matrix for fused sparse group
4 % lasso penalties. Provides input for fsglfit.m.
5 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
6 %%% INPUTS
7 % dim1: dimension 1 (x) of 3D image volume
8 % dim2: dimension 2 (y) of 3D image volume
9 % dim3: dimension 3 (z) of 3D image volume
10 % mask: vector of length dim1*dim2*dim3 that has a 1 for each predictor
11 % voxel in the 3D volume and a 0 for each voxel that is not
12 % of interest in the 3D volume (e.g. voxels outside of brain).
13 % The sum of the mask vector equals to p, the number of predictors.
14 % NOTE: The 3D volume is assumed to be concatenated into a vector
15 % in the same way the reshape function works in Matlab, i.e.,
16 % counting fastest along dim1, then dim2, then dim3.
17 % groups: vector of length p that assigns each predictor to a unique
18 % group, using integers 1, 2, etc.
19 %%% OUTPUTS
20 % K: K matrix
21 % (input for fsglfit_adaptive function)
22 % nj: number of predictors, p
23 % (input for fsglfit_adaptive function)
24 % nd: number of rows of D matrix
25 % (input for fsglfit_adaptive function)
26 % ngroups: number of coefficient groups
27 % (input for fsglfit_adaptive function)
28 % groupsizes: vector of coefficient group sizes
29 % (input for fsglfit_adaptive function)
30 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
31
32 function [K, nj, nd, ngroups, groupsizes] = makeKmatrix(dim1, dim2, dim3, mask, groups)
33 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
34 % make J matrix for lasso penalty
35 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
36 masksize = sum(mask);
```

```

37 J = sparse(1:masksize, 1:masksize, repelem(1, masksize), masksize, masksize);
38 % save the number of rows of J
39 nj = masksize;
40 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
41 % make D matrix for fused penalty
42 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
43 Dnrow = 3*dim1*dim2*dim3 - dim3*dim2 - dim3*dim1 - dim1*dim2;
44 Dncol = dim1*dim2*dim3;
45 % create column indices
46 % ONES
47 % to bind columns
48 bindcolones = 1:dim1*dim2*dim3;
49 bindcolonesremove = (1:(dim2*dim3))*dim1;
50 bindcolones(bindcolonesremove) = [];
51 % to bind rows
52 bindrowones = 1:dim1*dim2*dim3;
53 a = (1:dim1) + (dim1*(dim2-1));
54 A = repmat(a', 1, dim3);
55 b = [0, (1:(dim3-1)) * (dim1*dim2)];
56 c = bsxfun(@plus, A, b);
57 bindrowonesremove = reshape(c, [1, numel(c)]);
58 bindrowones(bindrowonesremove) = [];
59 % to bind slices
60 bindsliceones = 1:dim1*dim2*dim3;
61 bindsliceonesremove = ((dim1*dim2*dim3 - dim1*dim2) + 1):dim1*dim2*dim3;
62 bindsliceones(bindsliceonesremove) = [];
63 % NEGATIVE ONES
64 bindcolnegones = bindcolones + 1;
65 bindrownegones = bindrowones + dim1;
66 bindslicenegones = bindsliceones + (dim1*dim2);
67 % put all column indices together
68 Dcolindices = [bindcolones bindrowones bindsliceones bindcolnegones bindrownegones bindslicenegones];
69 % make unmasked D matrix
70 Dunmasked = sparse([1:Dnrow 1:Dnrow], Dcolindices, [repelem(1, Dnrow) repelem(-1, Dnrow)], Dnrow, Dncol);
71 % clear some temporary variables
72 clear bindcolones bindcolnegones bindcolonesremove bindrowones bindrowneg ones bindrowonesremove bindsliceones bindslicenegones
    bindsliceonesremove a b c Dncol Dnrow
73 % apply the mask to the D matrix
74 % remove any row that has an entry at a zero position in the mask
75 % then remove all columns corresponding to zero positions in the mask
76 Dmaskzerocols = abs(Dunmasked(:,mask==0));
77 Dmaskzerocolsrowsum = sum(Dmaskzerocols, 2);
78 % tabulate(Dmaskedcolsrowsum)
79 D = Dunmasked(Dmaskzerocolsrowsum==0, mask==1);
80 % save rows of D
81 nd = size(D, 1);
82 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
83 % make G matrix for groups
84 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
85 % for a complete partition of the data as in this case
86 % the G matrix is just a reordering of the rows of the J matrix
87 grouptab = tabulate(groups);
88 % save the number of groups
89 ngroups = size(grouptab, 1);
90 % save the number of voxels in each group;
91 groupsizes = grouptab(:,2);
92 % create an ordering of the indices
93 [~, groupindices] = sort(groups);
94 % make the G matrix
95 G = sparse(1:masksize, groupindices, repelem(1, masksize), masksize, masksize);
96 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
97 % combine J D and G to form K
98 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
99 K = [J; D; G];
100 end

```

E.2 MATLAB FUNCTION: makeKmatrix_adaptive.m

```

1  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2  % MAKEKMATRIX_ADAPTIVE
3  % Function to create K matrix and generate adaptive
4  % weights for fused sparse group lasso adaptive
5  % penalties. Provides input for fsglfit_adaptive.m.
6  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
7  %%% INPUTS
8  % dim1: dimension 1 (x) of 3D image volume
9  % dim2: dimension 2 (y) of 3D image volume
10 % dim3: dimension 3 (z) of 3D image volume
11 % mask: vector of length dim1*dim2*dim3 that has a 1 for each predictor
12 % voxel in the 3D volume and a 0 for each voxel that is not
13 % of interest in the 3D volume (e.g. voxels outside of brain).
14 % The sum of the mask vector equals to p, the number of predictors.
15 % NOTE: The 3D volume is assumed to be concatenated into a vector
16 % in the same way the reshape function works in Matlab, i.e.,
17 % counting fastest along dim1, then dim2, then dim3.
18 % groups: vector of length p that assigns each predictor to a unique
19 % group, using integers 1, 2, etc.
20 % betaestimates: predetermined estimates for beta, e.g. from OLS or
21 % ridge regression results, used for defining the adaptive weights
22 %%% OUTPUTS
23 % K: K matrix
24 % (input for fsglfit_adaptive function)
25 % nj: number of predictors, p
26 % (input for fsglfit_adaptive function)
27 % nd: number of rows of D matrix
28 % (input for fsglfit_adaptive function)
29 % ngroups: number of coefficient groups
30 % (input for fsglfit_adaptive function)
31 % groupsizes: vector of coefficient group sizes
32 % (input for fsglfit_adaptive function)
33 % weights: adaptive penalty weights to rescale lambda
34 % (input for fsglfit_adaptive function)
35 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
36
37 function [K, nj, nd, ngroups, groupsizes, weights] = ...
38     makeKmatrix(dim1, dim2, dim3, mask, groups, betaestimates)
39 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
40 % make J matrix for lasso penalty
41 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
42 masksize = sum(mask);
43 J = sparse(1:masksize, 1:masksize, repelem(1, masksize), masksize, masksize);
44 % save the number of rows of J
45 nj = masksize;
46 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
47 % make D matrix for fused penalty
48 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
49 Dnrow = 3*dim1*dim2*dim3 - dim3*dim2 - dim3*dim1 - dim1*dim2;
50 Dncol = dim1*dim2*dim3;
51 % create column indices
52 % ONES
53 % to bind columns
54 bindcolones = 1:dim1*dim2*dim3;
55 bindcolonesremove = (1:(dim2*dim3))*dim1;
56 bindcolones(bindcolonesremove) = [];
57 % to bind rows
58 bindrowones = 1:dim1*dim2*dim3;
59 a = (1:dim1) + (dim1*(dim2-1));
60 A = repmat(a', 1, dim3);
61 b = [0, (1:(dim3-1)) * (dim1*dim2)];
62 c = bsxfun(@plus, A, b);
63 bindrowonesremove = reshape(c, [1, numel(c)]);
64 bindrowones(bindrowonesremove) = [];
65 % to bind slices
66 bindsliceones = 1:dim1*dim2*dim3;
67 bindsliceonesremove = ((dim1*dim2*dim3 - dim1*dim2) + 1):dim1*dim2*dim3;
68 bindsliceones(bindsliceonesremove) = [];
69 % NEGATIVE ONES
70 bindcolnegones = bindcolones + 1;
71 bindrownegones = bindrowones + dim1;
72 bindslicenegones = bindsliceones + (dim1*dim2);
73 % put all column indices together
74 Dcolindices = [bindcolones bindrowones bindsliceones bindcolnegones bindrownegones bindslicenegones];
75 % make unmasked D matrix

```

```

76 Dunmasked = sparse([1:Dnrow 1:Dnrow], Dcolindices, [repelem(1, Dnrow) repelem(-1, Dnrow)], Dnrow, Dncol);
77 % clear some temporary variables
78 clear bindcolones bindcolnegones bindcolonesremove bindrowones bindrowneg ones bindrowonesremove bindsliceones bindslicenegones
    bindsliceonesremove a A b c Dncol Dnrow
79 % apply the mask to the D matrix
80 % remove any row that has an entry at a zero position in the mask
81 % then remove all columns corresponding to zero positions in the mask
82 Dmaskzerocols = abs(Dunmasked(:,mask==0));
83 Dmaskzerocolsrowsum = sum(Dmaskzerocols, 2);
84 % tabulate(Dmaskedcolsrowsum)
85 D = Dunmasked(Dmaskzerocolsrowsum==0, mask==1);
86 % save rows of D
87 nd = size(D, 1);
88 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
89 % make G matrix for groups
90 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
91 % for a complete partition of the data as in this case
92 % the G matrix is just a reordering of the rows of the J matrix
93 grouptab = tabulate(groups);
94 % save the number of groups
95 ngroups = size(grouptab, 1);
96 % save the number of voxels in each group;
97 groupsizes = grouptab(:,2);
98 % create an ordering of the indices
99 [~, groupindices] = sort(groups);
100 % make the G matrix
101 G = sparse(1:masksize, groupindices, repelem(1, masksize), masksize, masksize);
102 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
103 % combine J D and G to form K
104 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
105 K = [J; D; G];
106 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
107 % make the weight vector
108 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
109 % calculate lasso weights
110 % take absolute value
111 absbetaestimates = abs(betaestimates);
112 % take inverse
113 absbetaestimatesinv = 1./absbetaestimates;
114 % rescale so coefficients add to p
115 lassowts = absbetaestimatesinv*(nj/sum(absbetaestimatesinv));
116 % calculate fusion weights
117 fusionwts = (nj/sum(absbetaestimatesinv))./(abs(D * betaestimates));
118 % calculate group weights
119 groupwts = zeros(ngroups, 1);
120 for i = 1:ngroups
121     groupwts(i) = (nj/sum(absbetaestimatesinv))./norm(betaestimates(groups==i));
122 end
123 weights = [lassowts; fusionwts; groupwts];
124 end

```

E.3 MATLAB FUNCTION: fsglfit.m

```

1  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2  % FSGLFIT
3  % Function to fit fused sparse group lasso
4  % for non-adaptive penalties.
5  % Uses output from function makeKmatrix.m.
6  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
7  %%% INPUTS
8  % X: n*p predictor matrix
9  % Y: n*1 scalar outcome
10 % K: K matrix
11 % (output from makeKmatrix function)
12 % nj: number of predictors, p
13 % (output from makeKmatrix function)
14 % nd: number of rows of D matrix
15 % (output from makeKmatrix function)
16 % ngroups: number of coefficient groups
17 % (output from makeKmatrix function)
18 % groupsizes: vector of coefficient group sizes
19 % (output from makeKmatrix function)
20 % alpha: tuning parameter that controls balance between group (alpha = 0)
21 % and L1 lasso (alpha = 1) penalty terms
22 % gamma: tuning parameter that controls balance between fusion (gamma = 0)
23 % and sparsity (gamma = 1) penalty terms
24 % lambda: tuning parameter that controls overall level of regularization
25 % rho: initial ADMM step-size
26 % Niter: maximum number of ADMM iterations
27 %%% OUTPUTS
28 % pred_y: predicted Y values
29 % mse_y: mean squared error of the predicted Y values
30 % beta: final value for beta (estimated beta)
31 % theta: final value for theta
32 % mu: final value for mu
33 % niter: number of iterations before stopping
34 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
35
36 function [pred_y, mse_y, beta, theta, mu, niter]...
37     = fsglfit(...
38     X,...
39     Y,...
40     K,...
41     nj,...
42     nd,...
43     ngroups,...
44     groupsizes,...
45     alpha,...
46     gamma,...
47     lambda,...
48     rho,...
49     Niter)
50 % define soft threshold function
51 function b = softthresh(a, kappa)
52     if sum(a(:)==0) == size(a(:))
53         b = a;
54     else
55         b = max(0, (1 - kappa/norm(a))) * a;
56     end
57 end
58 % define epsilons
59 % NOTE: These can be changed to match desired tolerance
60 epsilon_abs = 10 ^ -3;
61 epsilon_rel = 10 ^ -3;
62 % center columns of X and Y
63 Xcentered = detrend(X, 'constant');
64 Ycentered = detrend(Y, 'constant');
65 % make a cell array with indices of each group for theta update
66 ni = [repelem(1, nj) repelem(1, nd) groupsizes]; % vector of group sizes
67 cumsumni = [0 cumsum(ni)];
68 N = nj + nd + ngroups;
69 groupindices = cell(1, N);
70 for i = 2:(N + 1)
71     groupindices{i-1} = (cumsumni(i-1) + 1):cumsumni(i);
72 end
73 % calculate beta update factor
74 % which does not change over loop iterations
75 beta_update_factor = Xcentered' * Xcentered + rho * (K' * K);

```

```

76 % calculate beta update term
77 % which does not change over loop iterations
78 beta_update_term1 = Xcentered' * Ycentered;
79 % set initial values to zero
80 beta0 = repelem(0, size(K, 2));
81 theta0 = repelem(0, size(K, 1));
82 mu0 = repelem(0, size(K, 1));
83 % initialize parameters
84 currentbeta = beta0';
85 currenttheta = theta0';
86 currentmu = mu0';
87 clear beta0 theta0 mu0
88 % define lambda vector
89 Lambda = [...
90     repelem(alpha * gamma * lambda, nj)...
91     repelem((1 - gamma) * lambda, nd)...
92     repelem((1 - alpha) * gamma * lambda, ngroups)];
93 % LOOP over ADMM iterations
94 for t = 1:Niter
95     % UPDATE PARAMETERS
96     % update beta
97     beta_update_term2 = K' * (currentmu - rho * currenttheta);
98     currentbeta = beta_update_factor \ (beta_update_term1 - beta_update_term2);
99     % update theta
100     eta = K * currentbeta + currentmu / rho;
101     previoustheta = currenttheta;
102     for i = 1:N
103         currenttheta(groupindices{i}) = softthresh(eta(groupindices{i}),...
104             (Lambda(i) * sqrt(numel(eta(groupindices{i})))) / rho);
105     end
106     % update mu
107     currentmu = currentmu + rho * (K * currentbeta - currenttheta);
108     % CHECK STOPPING CONDITIONS
109     % calculate dual residual
110     epsilon_dual = sqrt(size(currenttheta, 1)) * epsilon_abs...
111         + epsilon_rel * norm(K' * currentmu);
112     current_s = norm(rho * K' * (previoustheta - currenttheta));
113     % calculate primal residual
114     epsilon_primal = sqrt(size(currentbeta, 1)) * epsilon_abs...
115         + epsilon_rel * max(norm(K * currentbeta), norm(currenttheta));
116     current_r = norm(K * currentbeta - currenttheta);
117     % break loop if stopping conditions are met
118     if (current_s < epsilon_dual) && (current_r < epsilon_primal)
119         break
120     end
121     % update rho
122     if (current_r > 10 * current_s)
123         rho = 2 * rho;
124     elseif (current_s > 10 * current_r)
125         rho = rho / 2;
126     else rho = rho;
127     end
128 end % end loop over ADMM iterations
129 % save outputs
130 pred_y = Xcentered * currentbeta;
131 mse_y = mean((Ycentered - pred_y).^2);
132 beta = currentbeta;
133 theta = currenttheta;
134 mu = currentmu;
135 niter = t;
136 end

```


E.4 MATLAB FUNCTION: fsglfit_adaptive.m

```

1  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2  % FSGLFIT_ADAPTIVE
3  % Function to fit fused sparse group lasso
4  % for adaptive penalties.
5  % Uses output from function makeKmatrix_adaptive.m.
6  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
7  %%% INPUTS
8  % X: n*p predictor matrix
9  % Y: n*1 scalar outcome
10 % K: K matrix
11 % (output from makeKmatrix_adaptive function)
12 % nj: number of predictors, p
13 % (output from makeKmatrix_adaptive function)
14 % nd: number of rows of D matrix
15 % (output from makeKmatrix_adaptive function)
16 % ngroups: number of coefficient groups
17 % (output from makeKmatrix_adaptive function)
18 % groupsizes: vector of coefficient group sizes
19 % (output from makeKmatrix_adaptive function)
20 % weights: adaptive penalty weights to rescale lambda
21 % (output from makeKmatrix_adaptive function)
22 % initialbeta: initial coefficient values
23 % alpha: tuning parameter that controls balance between group (alpha = 0)
24 % and L1 lasso (alpha = 1) penalty terms
25 % gamma: tuning parameter that controls balance between fusion (gamma = 0)
26 % and sparsity (gamma = 1) penalty terms
27 % lambda: tuning parameter that controls overall level of regularization
28 % rho: initial ADMM step-size
29 % Niter: maximum number of ADMM iterations
30 %%% OUTPUTS
31 % pred_y: predicted Y values
32 % mse_y: mean squared error of the predicted Y values
33 % beta: final value for beta (estimated beta)
34 % theta: final value for theta
35 % mu: final value for mu
36 % niter: number of iterations before stopping
37 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
38
39 function [pred_y, mse_y, beta, theta, mu, niter]...
40     = fsglfit(...
41     X,...
42     Y,...
43     K,...
44     nj,...
45     nd,...
46     ngroups,...
47     groupsizes,...
48     weights,...
49     initialbeta,...
50     alpha,...
51     gamma,...
52     lambda,...
53     rho,...
54     Niter)
55 % define soft threshold function
56 function b = softthresh(a, kappa)
57     if sum(a(:)==0) == size(a(:))
58         b = a;
59     else
60         b = max(0, (1 - kappa/norm(a))) * a;
61     end
62 end
63 % define epsilons
64 % NOTE: These can be changed to match desired tolerance
65 epsilon_abs = 10 ^ -3;
66 epsilon_rel = 10 ^ -3;
67 % center columns of X and Y
68 Xcentered = detrend(X, 'constant');
69 Ycentered = detrend(Y, 'constant');
70 % make a cell array with indices of each group for theta update
71 ni = [repelem(1, nj) repelem(1, nd) groupsizes]; % vector of group sizes
72 cumsumni = [0 cumsum(ni)];
73 N = nj + nd + ngroups;
74 groupindices = cell(1, N);
75 for i = 2:(N + 1)

```

```

76     groupindices{i-1} = (cumsumni(i-1) + 1):cumsumni(i);
77 end
78 % calculate beta update factor
79 % which does not change over loop iterations
80 beta_update_factor = Xcentered' * Xcentered + rho * (K' * K);
81 % calculate beta update term
82 % which does not change over loop iterations
83 beta_update_term1 = Xcentered' * Ycentered;
84 % set initial values
85 beta0 = initialbeta;
86 theta0 = repelem(0, size(K, 1));
87 mu0 = repelem(0, size(K, 1));
88 % initialize parameters
89 currentbeta = beta0';
90 currenttheta = theta0';
91 currentmu = mu0';
92 clear beta0 theta0 mu0
93 % define lambda vector
94 Lambda = [...
95     repelem(alpha * gamma * lambda, nj)...
96     repelem((1 - gamma) * lambda, nd)...
97     repelem((1 - alpha) * gamma * lambda, ngroups)];
98 % LOOP over ADMM iterations
99 for t = 1:Niter
100     % UPDATE PARAMETERS
101     % update beta
102     beta_update_term2 = K' * (currentmu - rho * currenttheta);
103     currentbeta = beta_update_factor \ (beta_update_term1 - beta_update_term2);
104     % update theta
105     eta = K * currentbeta + currentmu / rho;
106     previoustheta = currenttheta;
107     for i = 1:N
108         currenttheta(groupindices{i}) = softthresh(eta(groupindices{i}),...
109             (Lambda(i) * weights(i)) / rho);
110     end
111     % update mu
112     currentmu = currentmu + rho * (K * currentbeta - currenttheta);
113     % CHECK STOPPING CONDITIONS
114     % calculate dual residual
115     epsilon_dual = sqrt(size(currenttheta, 1)) * epsilon_abs...
116         + epsilon_rel * norm(K' * currentmu);
117     current_s = norm(rho * K' * (previoustheta - currenttheta));
118     % calculate primal residual
119     epsilon_primal = sqrt(size(currentbeta, 1)) * epsilon_abs...
120         + epsilon_rel * max(norm(K * currentbeta), norm(currenttheta));
121     current_r = norm(K * currentbeta - currenttheta);
122     % break loop if stopping conditions are met
123     if (current_s < epsilon_dual) && (current_r < epsilon_primal)
124         break
125     end
126     % update rho
127     if (current_r > 10 * current_s)
128         rho = 2 * rho;
129     elseif (current_s > 10 * current_r)
130         rho = rho / 2;
131     else rho = rho;
132     end
133 end % end loop over ADMM iterations
134 % save outputs
135 pred_y = Xcentered * currentbeta;
136 mse_y = mean((Ycentered - pred_y).^2);
137 beta = currentbeta;
138 theta = currenttheta;
139 mu = currentmu;
140 niter = t;
141 end

```

BIBLIOGRAPHY

- Abraham, A., Milham, M. P., Di Martino, A., Craddock, R. C., Samaras, D., Thirion, B. & Varoquaux, G. (2017), ‘Deriving reproducible biomarkers from multi-site resting-state data: An autism-based example’, *NeuroImage* **147**, 736–745.
- Andrews-Hanna, J. R., Snyder, A. Z., Vincent, J. L., Lustig, C., Head, D., Raichle, M. E. & Buckner, R. L. (2007), ‘Disruption of large-scale brain systems in advanced aging’, *Neuron* **56**(5), 924–935.
- Arbabshirani, M. R., Plis, S., Sui, J. & Calhoun, V. D. (2017), ‘Single subject prediction of brain disorders in neuroimaging: Promises and pitfalls’, *NeuroImage* **145**, 137–165.
- Ashburner, J., Barnes, G., Chen, C., Daunizeau, J., Flandin, G., Friston, K., Kiebel, S., Kilner, J., Litvak, V., Moran, R. et al. (2014), ‘SPM12 manual’, *Wellcome Trust Centre for Neuroimaging, London, UK*.
- Ashburner, J. & Friston, K. J. (2005), ‘Unified segmentation’, *NeuroImage* **26**(3), 839–851.
- Bach, F., Jenatton, R., Mairal, J., Obozinski, G. et al. (2012), ‘Structured sparsity through convex optimization’, *Statistical Science* **27**(4), 450–468.
- Baio, J. (2012), ‘Prevalence of autism spectrum disorders: Autism and Developmental Disabilities Monitoring Network, 14 Sites, United States, 2008. Morbidity and Mortality Weekly Report. Surveillance Summaries. Volume 61, Number 3.’, *Centers for Disease Control and Prevention*.
- Baldassarre, L., Mourao-Miranda, J. & Pontil, M. (2012), Structured sparsity models for brain decoding from fMRI data, in ‘Pattern Recognition in NeuroImaging (PRNI), 2012 International Workshop on’, IEEE, pp. 5–8.
- Bandettini, P. A. (2012), ‘Twenty years of functional MRI: The science and the stories’, *NeuroImage* **62**(2), 575–588.
- Bland, J. M. & Altman, D. G. (2011), ‘Correlation in restricted ranges of data’, *BMJ* **342**, d556.

- Boyd, S., Parikh, N., Chu, E., Peleato, B. & Eckstein, J. (2011), ‘Distributed optimization and statistical learning via the alternating direction method of multipliers’, *Foundations and Trends® in Machine Learning* **3**(1), 1–122.
- Calhoun, V. D. & Sui, J. (2016), ‘Multimodal fusion of brain imaging data: A key to finding the missing link(s) in complex mental illness’, *Biological psychiatry: Cognitive neuroscience and neuroimaging* **1**(3), 230–244.
- Cerliani, L., Mennes, M., Thomas, R. M., Di Martino, A., Thioux, M. & Keyzers, C. (2015), ‘Increased functional connectivity between subcortical and cortical resting-state networks in autism spectrum disorder’, *JAMA psychiatry* **72**(8), 767–777.
- Chu, C., Hsu, A.-L., Chou, K.-H., Bandettini, P., Lin, C., Initiative, A. D. N. et al. (2012), ‘Does feature selection improve classification accuracy? Impact of sample size and feature selection on classification using anatomical magnetic resonance images’, *NeuroImage* **60**(1), 59–70.
- Cohen, J. R., Asarnow, R. F., Sabb, F. W., Bilder, R. M., Bookheimer, S. Y., Knowlton, B. J. & Poldrack, R. A. (2011), ‘Decoding continuous variables from neuroimaging data: Basic and clinical applications’, *Frontiers in neuroscience* **5**.
- Cole, J. H., Marioni, R. E., Harris, S. E. & Deary, I. J. (2018), ‘Brain age and other bodily ‘ages’: Implications for neuropsychiatry’, *Molecular psychiatry* p. 1.
- Cole, J. H., Poudel, R. P., Tsagkrasoulis, D., Caan, M. W., Steves, C., Spector, T. D. & Montana, G. (2017), ‘Predicting brain age with deep learning from raw imaging data results in a reliable and heritable biomarker’, *NeuroImage* **163**, 115–124.
- Cole, J. H., Ritchie, S. J., Bastin, M. E., Hernández, M. V., Maniega, S. M., Royle, N., Corley, J., Pattie, A., Harris, S. E., Zhang, Q. et al. (2017), ‘Brain age predicts mortality’, *Molecular psychiatry* .
- Constantino, J. N., Davis, S. A., Todd, R. D., Schindler, M. K., Gross, M. M., Brophy, S. L., Metzger, L. M., Shoushtari, C. S., Splinter, R. & Reich, W. (2003), ‘Validation of a brief quantitative measure of autistic traits: Comparison of the social responsiveness scale with the autism diagnostic interview-revised’, *Journal of autism and developmental disorders* **33**(4), 427–433.
- Cox, R. W. (1996), ‘AFNI: Software for analysis and visualization of functional magnetic resonance neuroimages’, *Computers and Biomedical research* **29**(3), 162–173.
- Craddock, R., Benhajali, Y., Chu, C., Chouinard, F., Evans, A., Jakab, A., Khundrakpam, B. S., Lewis, J. D., Li, Q., Milham, M. et al. (2013), ‘The Neuro Bureau Preprocessing Initiative: Open sharing of preprocessed neuroimaging data and derivatives’, *Frontiers in Neuroinformatics (Neuroinformatics 2013)* .

- Cui, Z. & Gong, G. (2018), ‘The effect of machine learning regression algorithms and sample size on individualized behavioral prediction with functional connectivity features’, *NeuroImage* .
- Damoiseaux, J., Beckmann, C., Arigita, E. S., Barkhof, F., Scheltens, P., Stam, C., Smith, S. & Rombouts, S. (2007), ‘Reduced resting-state brain activity in the ‘default network’ in normal aging’, *Cerebral cortex* **18**(8), 1856–1864.
- Di Martino, A., Yan, C.-G., Li, Q., Denio, E., Castellanos, F. X., Alaerts, K., Anderson, J. S., Assaf, M., Bookheimer, S. Y., Dapretto, M. et al. (2014), ‘The Autism Brain Imaging Data Exchange: Towards a large-scale evaluation of the intrinsic brain architecture in autism’, *Molecular psychiatry* **19**(6), 659.
- Eavani, H., Habes, M., Satterthwaite, T. D., An, Y., Hsieh, M.-K., Honnorat, N., Erus, G., Doshi, J., Ferrucci, L., Beason-Held, L. L. et al. (2018), ‘Heterogeneity of structural and functional imaging patterns of advanced brain aging revealed via machine learning methods’, *Neurobiology of Aging* .
- Eloyan, A., Muschelli, J., Nebel, M. B., Liu, H., Han, F., Zhao, T., Barber, A. D., Joel, S., Pekar, J. J., Mostofsky, S. H. et al. (2012), ‘Automated diagnoses of attention deficit hyperactive disorder using magnetic resonance imaging’, *Frontiers in systems neuroscience* **6**.
- Fiot, J.-B., Raguet, H., Risser, L., Cohen, L. D., Fripp, J., Vialard, F.-X., Initiative, A. D. N. et al. (2014), ‘Longitudinal deformation models, spatial regularizations and learning strategies to quantify Alzheimer’s disease progression’, *NeuroImage: Clinical* **4**, 718–729.
- Fjell, A. M. & Walhovd, K. B. (2010), ‘Structural brain changes in aging: Courses, causes and cognitive consequences’, *Reviews in the Neurosciences* **21**(3), 187–222.
- Franke, K. & Gaser, C. (2012), ‘Longitudinal changes in individual brainAGE in healthy aging, mild cognitive impairment, and Alzheimer’s disease’, *GeroPsych* .
- Franke, K., Ziegler, G., Klöppel, S., Gaser, C., Initiative, A. D. N. et al. (2010), ‘Estimating the age of healthy subjects from T1-weighted MRI scans using kernel methods: Exploring the influence of various parameters’, *NeuroImage* **50**(3), 883–892.
- Friedman, J., Hastie, T. & Tibshirani, R. (2010), ‘Regularization paths for generalized linear models via coordinate descent’, *Journal of Statistical Software* **33**(1), 1–22.
URL: <http://www.jstatsoft.org/v33/i01/>
- Gaser, C., Franke, K., Klöppel, S., Koutsouleris, N., Sauer, H., Initiative, A. D. N. et al. (2013), ‘BrainAGE in mild cognitive impaired patients: Predicting the conversion to Alzheimer’s disease’, *PloS one* **8**(6), e67346.

- Grady, C. L., Springer, M. V., Hongwanishkul, D., McIntosh, A. R. & Winocur, G. (2006), ‘Age-related changes in brain activity across the adult lifespan’, *Journal of cognitive neuroscience* **18**(2), 227–241.
- Gramfort, A., Thirion, B. & Varoquaux, G. (2013), Identifying predictive regions from fMRI with TV-L1 prior, in ‘Pattern Recognition in Neuroimaging (PRNI), 2013 International Workshop on’, IEEE, pp. 17–20.
- Grosenick, L., Klingenberg, B., Katovich, K., Knutson, B. & Taylor, J. E. (2013), ‘Interpretable whole-brain prediction analysis with GraphNet’, *NeuroImage* **72**, 304–321.
- Hand, D. J. (2006), ‘Classifier technology and the illusion of progress’, *Statistical science* pp. 1–14.
- He, B., Yang, H. & Wang, S. (2000), ‘Alternating direction method with self-adaptive penalty parameters for monotone variational inequalities’, *Journal of Optimization Theory and applications* **106**(2), 337–356.
- Hinrichs, C., Singh, V., Xu, G., Johnson, S. C., Initiative, A. D. N. et al. (2011), ‘Predictive markers for AD in a multi-modality framework: An analysis of MCI progression in the ADNI population’, *NeuroImage* **55**(2), 574–589.
- Hoerl, A. E. & Kennard, R. W. (1970), ‘Ridge regression: Biased estimation for nonorthogonal problems’, *Technometrics* **12**(1), 55–67.
- Huo, Z. & Tseng, G. (2017), ‘Integrative sparse k-means with overlapping group lasso in genomic applications for disease subtype discovery’, *The annals of applied statistics* **11**(2), 1011.
- Insel, T., Cuthbert, B., Garvey, M., Heinssen, R., Pine, D. S., Quinn, K., Sanislow, C. & Wang, P. (2010), ‘Research domain criteria (RDoC): Toward a new classification framework for research on mental disorders’.
- Jack Jr, C. R., Knopman, D. S., Jagust, W. J., Petersen, R. C., Weiner, M. W., Aisen, P. S., Shaw, L. M., Vemuri, P., Wiste, H. J., Weigand, S. D. et al. (2013), ‘Update on hypothetical model of Alzheimer’s disease biomarkers’, *Lancet neurology* **12**(2), 207.
- Jacob, L., Obozinski, G. & Vert, J.-P. (2009), Group lasso with overlap and graph lasso, in ‘Proceedings of the 26th annual international conference on machine learning’, ACM, pp. 433–440.
- Kapur, S., Phillips, A. G. & Insel, T. R. (2012), ‘Why has it taken so long for biological psychiatry to develop clinical tests and what to do about it?’, *Molecular psychiatry* **17**(12), 1174–1179.
- Karim, H. T., Tudorascu, D. L., Cohen, A., Price, J. C., Lopresti, B., Mathis, C., Klunk, W., Snitz, B. E. & Aizenstein, H. J. (2018), ‘Activity in the executive control circuit is

- positively associated with amyloid burden and negatively associated with education in cognitively healthy older adults', *Under review*.
- Kassraian-Fard, P., Matthis, C., Balsters, J. H., Maathuis, M. H. & Wenderoth, N. (2016), 'Promises, pitfalls, and basic guidelines for applying machine learning classifiers to psychiatric imaging data, with autism as an example', *Frontiers in psychiatry* **7**, 177.
- Liem, F., Varoquaux, G., Kynast, J., Beyer, F., Masouleh, S. K., Huntenburg, J. M., Lampe, L., Rahim, M., Abraham, A., Craddock, R. C. et al. (2017), 'Predicting brain-age from multimodal imaging data captures cognitive impairment', *NeuroImage* **148**, 179–188.
- Liu, M., Zhang, D., Shen, D., Initiative, A. D. N. et al. (2014), 'Identifying informative imaging biomarkers via tree structured sparse learning for AD diagnosis', *Neuroinformatics* **12**(3), 381.
- Liu, S., Cai, W., Liu, S., Zhang, F., Fulham, M., Feng, D., Pujol, S. & Kikinis, R. (2015), 'Multimodal neuroimaging computing: A review of the applications in neuropsychiatric disorders', *Brain informatics* **2**(3), 167–180.
- Lustig, C., Snyder, A. Z., Bhakta, M., O'Brien, K. C., McAvoy, M., Raichle, M. E., Morris, J. C. & Buckner, R. L. (2003), 'Functional deactivations: Change with age and dementia of the Alzheimer type', *Proceedings of the National Academy of Sciences* **100**(24), 14504–14509.
- MATLAB (2016), *version 9.1.0 (R2016b)*, The MathWorks Inc., Natick, Massachusetts.
- Michel, V., Gramfort, A., Varoquaux, G., Eger, E. & Thirion, B. (2011), 'Total variation regularization for fMRI-based prediction of behavior', *IEEE transactions on medical imaging* **30**(7), 1328–1340.
- Morris, J. C. (1993), 'The clinical dementia rating (CDR): Current version and scoring rules.', *Neurology*.
- Mwangi, B., Tian, T. S. & Soares, J. C. (2014), 'A review of feature reduction techniques in neuroimaging', *Neuroinformatics* **12**(2), 229–244.
- Nielsen, J. A., Zielinski, B. A., Fletcher, P. T., Alexander, A. L., Lange, N., Bigler, E. D., Lainhart, J. E. & Anderson, J. S. (2013), 'Multisite functional connectivity MRI classification of autism: ABIDE results', *Frontiers in human neuroscience* **7**, 599.
- Obozinski, G., Jacob, L. & Vert, J.-P. (2011), 'Group lasso with overlaps: The latent group lasso approach', *arXiv preprint arXiv:1110.0413*.
- Pelphrey, K. A., Shultz, S., Hudac, C. M. & Vander Wyk, B. C. (2011), 'Research review: Constraining heterogeneity: The social brain and its development in autism spectrum disorder', *Journal of Child Psychology and Psychiatry* **52**(6), 631–644.

- Plitt, M., Barnes, K. A. & Martin, A. (2015), ‘Functional connectivity classification of autism identifies highly predictive brain features but falls short of biomarker standards’, *NeuroImage: Clinical* **7**, 359–366.
- R Core Team (2017), *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria.
URL: <https://www.R-project.org/>
- Raichle, M. E., MacLeod, A. M., Snyder, A. Z., Powers, W. J., Gusnard, D. A. & Shulman, G. L. (2001), ‘A default mode of brain function’, *Proceedings of the National Academy of Sciences* **98**(2), 676–682.
- Raz, N., Lindenberger, U., Rodrigue, K. M., Kennedy, K. M., Head, D., Williamson, A., Dahle, C., Gerstorf, D. & Acker, J. D. (2005), ‘Regional brain changes in aging healthy adults: General trends, individual differences and modifiers’, *Cerebral cortex* **15**(11), 1676–1689.
- Rudin, L. I., Osher, S. & Fatemi, E. (1992), ‘Nonlinear total variation based noise removal algorithms’, *Physica D: Nonlinear Phenomena* **60**(1-4), 259–268.
- Schultz, A. P., Chhatwal, J. P., Huijbers, W., Hedden, T., Van Dijk, K. R., McLaren, D. G., Ward, A. M., Wigman, S. & Sperling, R. A. (2014), ‘Template based rotation: A method for functional connectivity analysis with a priori templates’, *NeuroImage* **102**, 620–636.
- Shimizu, Y., Yoshimoto, J., Toki, S., Takamura, M., Yoshimura, S., Okamoto, Y., Yamawaki, S. & Doya, K. (2015), ‘Toward probabilistic diagnosis and understanding of depression based on functional MRI data analysis with logistic group lasso’, *PloS one* **10**(5), e0123524.
- Simon, N., Friedman, J., Hastie, T. & Tibshirani, R. (2013), ‘A sparse-group lasso’, *Journal of Computational and Graphical Statistics* **22**(2), 231–245.
- Steffener, J., Habeck, C., O’Shea, D., Razlighi, Q., Bherer, L. & Stern, Y. (2016), ‘Differences between chronological and brain age are related to education and self-reported physical activity’, *Neurobiology of aging* **40**, 138–144.
- Tibshirani, R. (1996), ‘Regression shrinkage and selection via the lasso’, *Journal of the Royal Statistical Society. Series B (Methodological)* pp. 267–288.
- Tibshirani, R., Saunders, M., Rosset, S., Zhu, J. & Knight, K. (2005), ‘Sparsity and smoothness via the fused lasso’, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **67**(1), 91–108.
- Tomasi, D. & Volkow, N. D. (2012), ‘Aging and functional brain networks’, *Molecular psychiatry* **17**(5), 549.
- Varikuti, D. P., Genon, S., Sotiras, A., Schwender, H., Hoffstaedter, F., Patil, K. R., Jockwitz, C., Caspers, S., Moebus, S., Amunts, K. et al. (2018), ‘Evaluation of non-negative matrix factorization of grey matter in age prediction’, *NeuroImage* **173**, 394–410.

- Viallon, V., Lambert-Lacroix, S., Höfling, H. & Picard, F. (2013), ‘Adaptive generalized fused-lasso: Asymptotic properties and applications’.
URL: <https://hal.archives-ouvertes.fr/hal-00813281>
- Wang, H. & Leng, C. (2008), ‘A note on adaptive group lasso’, *Computational statistics & data analysis* **52**(12), 5277–5286.
- Wardlaw, J. M., Hernández, M. C. V. & Muñoz-Maniega, S. (2015), ‘What are white matter hyperintensities made of? relevance to vascular cognitive impairment’, *Journal of the American Heart Association* **4**(6), e001140.
- Woo, C.-W., Chang, L. J., Lindquist, M. A. & Wager, T. D. (2017), ‘Building better biomarkers: Brain models in translational neuroimaging’, *Nature neuroscience* **20**(3), 365–377.
- Xin, B., Kawahara, Y., Wang, Y. & Gao, W. (2014), Efficient generalized fused lasso and its application to the diagnosis of Alzheimer’s disease., *in* ‘AAAI’, pp. 2163–2169.
- Yang, Y. & Zou, H. (2015), ‘A fast unified algorithm for solving group-lasso penalize learning problems’, *Statistics and Computing* **25**(6), 1129–1141.
- Yarkoni, T. & Westfall, J. (2017), ‘Choosing prediction over explanation in psychology: Lessons from machine learning’, *Perspectives on Psychological Science* **12**(6), 1100–1122.
- Yesavage, J. A., Brink, T. L., Rose, T. L., Lum, O., Huang, V., Adey, M. & Leirer, V. O. (1982), ‘Development and validation of a geriatric depression screening scale: A preliminary report’, *Journal of psychiatric research* **17**(1), 37–49.
- Yuan, M. & Lin, Y. (2006), ‘Model selection and estimation in regression with grouped variables’, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **68**(1), 49–67.
- Zhang, D., Wang, Y., Zhou, L., Yuan, H., Shen, D., Initiative, A. D. N. et al. (2011), ‘Multimodal classification of Alzheimer’s disease and mild cognitive impairment’, *NeuroImage* **55**(3), 856–867.
- Zhao, P., Rocha, G. & Yu, B. (2009), ‘The composite absolute penalties family for grouped and hierarchical variable selection’, *The Annals of Statistics* pp. 3468–3497.
- Zhou, J., Chen, J. & Ye, J. (2011), ‘Malsar: Multi-task learning via structural regularization’, *Arizona State University* **21**.
- Zhou, J., Liu, J., Narayan, V. A. & Ye, J. (2012), Modeling disease progression via fused sparse group lasso, *in* ‘Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining’, ACM, pp. 1095–1103.
- Zou, H. (2006), ‘The adaptive lasso and its oracle properties’, *Journal of the American statistical association* **101**(476), 1418–1429.

Zou, H. & Hastie, T. (2005), ‘Regularization and variable selection via the elastic net’, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **67**(2), 301–320.